Oracle® Application Integration Architecture

Siebel CRM Integration Pack for Oracle Order Management: Order to Cash Implementation Guide

Release 3.1.1

E39434-01

December 2013

Documentation that describes the Siebel Customer Relationship Management (Siebel CRM) Integration Pack for Oracle Order Management pre-built integration implementation process, which provides a seamless and robust order-to-cash business process. This guide provides Order to Cash business process flows and the configuration steps needed to complete implementation.

ORACLE

Oracle Application Integration Architecture Siebel CRM Integration Pack for Oracle Order Management: Order to Cash Implementation Guide, Release 3.1.1

E39434-01

Primary Author: Oracle Corporation

Contents

4 Process Integration for Product Management

8 Shipping Charges Integration Flow

9 Credit Check Integration Flow

10 Payment Authorization Integration Flow

11 Process Integration for Order Management

12 Process Integration for Asset Management

15 Configuring the Order to Cash Pre-Built Integration

List of Examples

List of Tables

List of Figures

Preface

Welcome to the Siebel CRM Integration Pack for Oracle Order Management: Order to Cash Implementation Guide.

What's New in this Guide

- The Oracle Application Integration Architecture Installation and Upgrade Guide for Pre-Built Integrations is restructured into a general installation chapter with an individual configuration and deployment chapter for each pre-built integration.
- The term *process integration pack* is replaced with the term *pre-built integrations*.
- The implementation guides are restructured into two parts: design and set up.

  Part I - Design: This part provides functional overviews, activity diagrams, assumptions and constraints, and technical sequence diagrams and steps.

  Part II - Set up: This part provides prerequisites, data requirements, and configuration steps.
- Starting with this release, these integrations are no longer available:
  - Oracle CRM On Demand Integration Pack for JD Edwards EnterpriseOne: Lead to Order
  - Oracle Workforce Administration Integration Pack for PeopleSoft Human Resources

Common Oracle AIA Pre-Built Integration Guides

Oracle Application Integration Architecture Pre-Built Integrations 3.1.1 includes the following guides shared by all products delivered with this release:

- Oracle Application Integration Architecture Installation and Upgrade Guide for Pre-Built Integrations Release 11.5

  This guide provides an overview of the installation process, including how to install, configure, and deploy your pre-built integrations. The steps required to upgrade your pre-built integrations to the latest release are also provided.
- Oracle Application Integration Architecture Pre-Built Integrations 11.1: Utilities Guide

  This guide describes:
  - How to work with and configure Session Pool Manager (SPM), which 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.

  - How to deploy and configure the AIACompositeScheduler. This is a utility component that is used by pre-built integrations to schedule a service-oriented architecture (SOA) composite to be invoked at the specified time interval.

- Oracle Application Integration Architecture Pre-Built Integrations 11.5: Product-to-Guide Index

  The Product-to-Guide index lists the guides that provide information for each product delivered in this release.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Master Notes

Master notes provide a one-stop reference point for the following:

- Latest manuals
- Alerts
- Troubleshooting details
- FAQs
- Patching information
- AIA community link and more.

For master notes for the Siebel CRM Integration Pack for Oracle Order Management: Order to Cash, visit Master Note ID 1387275.2 on My Oracle Support (MOS): http://support.oracle.com.

Additional Resources

The following resources are also available:

- **Oracle Application Integration Architecture Foundation Pack:**

  Oracle AIA Pre-Built integrations require Foundation Pack 11.1.1.5.0 to be installed. Refer to the Foundation Pack documentation library on OTN to download the Foundation Pack guides at http://download.oracle.com/docs/cd/E21764_01/aia.htm.

- **Oracle Application Integration Architecture: Product-to-Guide Index:**

  Oracle Technology Network: http://www.oracle.com/technetwork/index.html

- **Known Issues and Workarounds:**

My Oracle Support: https://support.oracle.com/

- **Release Notes:**

  Oracle Technology Network:
  http://www.oracle.com/technetwork/index.html

- **Documentation updates:**

  Oracle Technology Network:
  http://www.oracle.com/technetwork/index.html

Part I

Understanding the Delivered Integrations

This part provides an overview of the Siebel CRM Integration Pack for Oracle Order Management: Order to Cash and discusses the process integrations included in the Order to Cash: Siebel CRM - EBS pre-built integration.

Part I includes the following chapters:

1

Siebel CRM Integration Pack for Oracle Order Management: Order to Cash

This chapter provides an overview of the Siebel Customer Relationship Management (Siebel CRM) Integration Pack for Oracle Order Management and discusses the order to cash business process flows and solution assumptions and constraints.

This chapter includes the following sections:

1.1 Siebel CRM Integration Pack for Oracle Order Management

The Siebel CRM Integration Pack for Oracle Order Management (Order to Cash: Siebel CRM - EBS) provides a seamless and robust order-to-cash business process. Leveraging the best front-office with the best back-office applications, this integration provides a solution that gives a streamlined, end-to-end Order to Cash business process, which in turn enables a faster time to market for new products and a faster time to revenue.

A quote or order created in Siebel Quote and Order Capture can generate an order in Oracle E-Business Suite (Oracle EBS) Order Management. The order-to-cash flow consists of master data flows and transactional flows. For master data, customer account data is synchronized bidirectionally, while product (item or bill of material [BOM]) data is synchronized from the Oracle EBS item master.

The transactional flows are Available to Promise (ATP), Credit Check, Payment Authorization, Shipping Charges, Quotes, and the Order integration flow.

The integration includes additional touch points between Siebel CRM and Oracle EBS that are required to enable this process: loading price lists and synchronizing assets.

The integration also provides the ability to invoke the Oracle Configurator from either a Siebel quote or an order. This approach eliminates the costly and complex maintenance of configuration rules in Siebel CRM. It also leverages a customer's existing investment in Oracle Configurator.

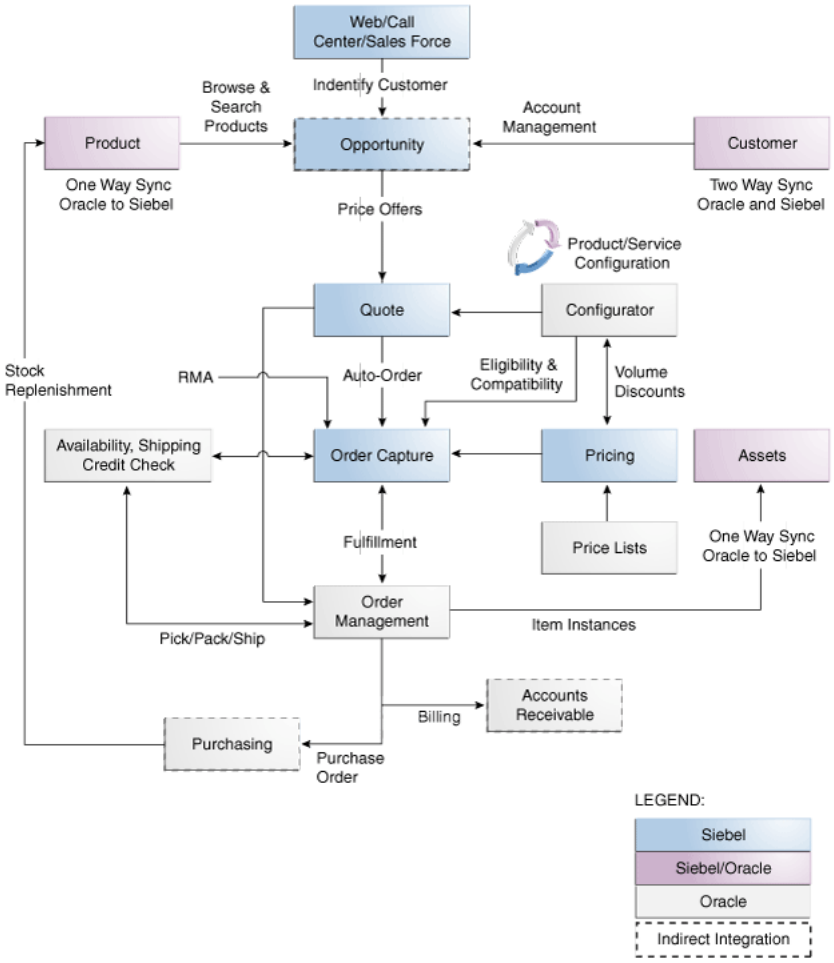
This business process helps organizations:

- Eliminate order fallout by ensuring that products are correctly configured and available before confirming the order.
- Increase customer satisfaction by ensuring that the order is priced correctly.

- Reduce customer callbacks by providing order availability and status.
- Decrease time to market for new products.
- Enable faster time to revenue.
- Reduce the implementation costs using delivered mappings and business processes.

Figure 1–1 illustrates the Order to Cash: Siebel CRM - EBS integration solution.

Figure 1–1 Order to Cash Integration Functional Diagram



1.2 Order to Cash Business Process Flows

The Order to Cash: Siebel CRM - EBS integration consists of these integration flows:

- Initial bulk data loading integration flows for customers, products, price lists, and assets.
- Customer Management.
- Product Management.
- Price List Management.
- Process integration for Quotes.

- Service calls: Available to Promise, Credit Check, Payment Authorization, and Shipping Charges.
- Order Management.
- Asset Management.

Figure 1–2 and Figure 1–3 illustrate the Order to Cash business process flow:

Figure 1–2 Order to Cash Business Process Flow (1 of 2)

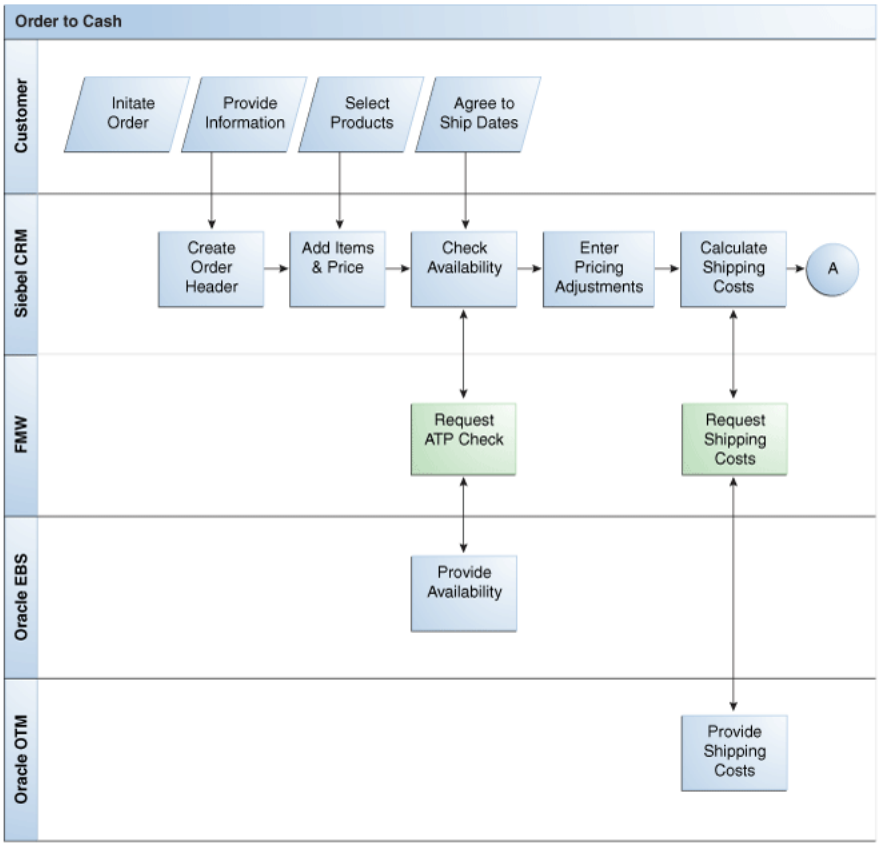
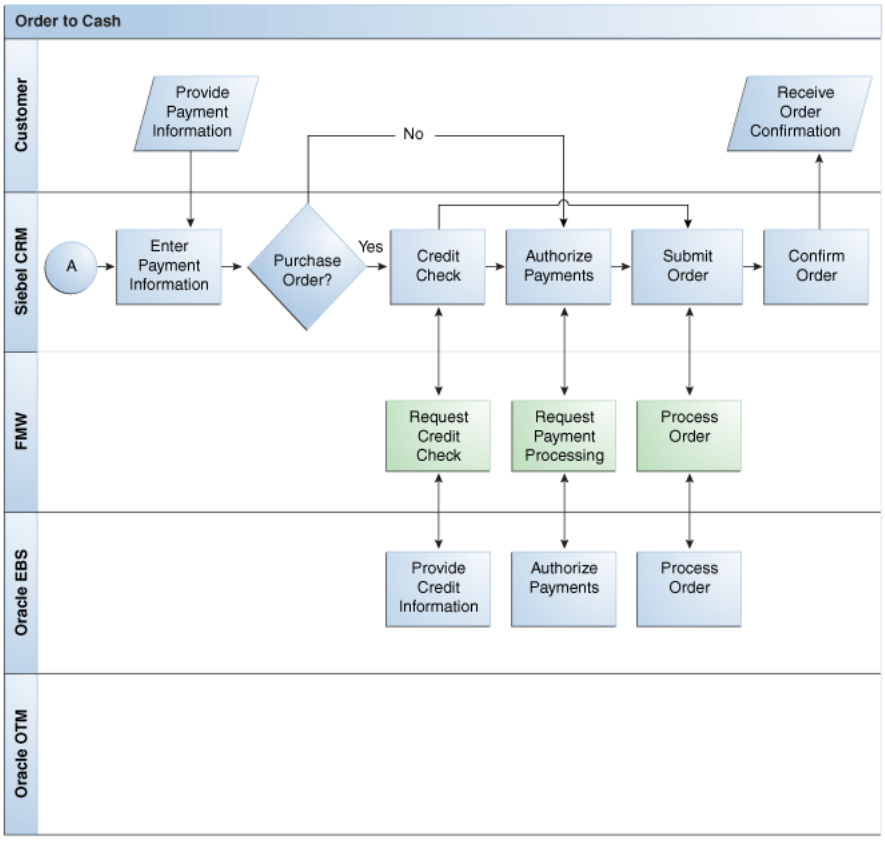


Figure 1–3 Order to Cash Business Process Flow (2 of 2)



The Order to Cash business process enables deployments to create and update an order across the life cycle of the entire order. This business process supports complex pricing and product configuration, availability checking, credit verification, shipping and tax calculations, and multiple methods of payment to ensure that before any orders delivered to the customer are complete and accurate. The initial data load processes enable bulk loading of customers, products, price lists, and assets.

In the Order to Cash process, customers contact customer service representatives (CSRs) to either place new orders or revise existing orders. The CSR first determines if the customer is new or exists in the system. For new customers, the CSR captures all of the pertinent information and creates an account in Siebel CRM. Next, the CSR creates the required orders. Working with the customer, the CSR selects products in order lines and enters shipping and billing information.

It is good practice to do the availability check to ensure that the customer's order can be fulfilled successfully in the back office (Available to Promise [ATP]). For new customers, or depending on the value of the order, the CSR should perform a credit check to ensure that the order is not placed on hold for credit reasons. It is also possible to calculate shipping charges before an order is submitted to the back office. If the ATP and credit checks are successful, the CSR can submit the order.

Order updates, including taxes and shipping charges due to order fulfillment in the back office, are synchronized to Siebel CRM. The CSR can access the latest order status, thereby serving customers better without having to toggle to a different application.

1.3 Order to Cash Solution Assumptions and Constraints

These are the assumptions and constraints for the Order to Cash: Siebel CRM - EBS pre-built integration:

1. Siebel Industry Applications (Siebel SIA) version is supported out-of-the-box.
2. Only Oracle Configurator is supported out-of-the-box.
3. Oracle EBS is the product master, and the item/BOM synchronization happens only from Oracle EBS to Siebel CRM.
4. Pricing is performed in Siebel CRM. Oracle EBS does not recalculate the price.
5. Business-to-customer (B2C) order scenarios are not supported out-of-the-box.
6. Prospects from Siebel are not synchronized to Oracle EBS.
7. Participating applications are implemented in either the same code page or Unicode.

Note: Additional assumptions and constraints exist for each of the process integration flows; they are listed in the respective chapters.

2

Loading Initial Data

This chapter provides an overview of initial bulk data loads, discusses how to deploy Oracle Data Integration (ODI) repository components and explains how to load initial customer data, product data, price list data, and assets data.

This chapter includes the following sections:

2.1 Understanding Initial Bulk Data Loads

The Order to Cash integration requires that the business data is always synchronized in both Oracle and Siebel databases. The Order to Cash integration enables you to:

- Initially load your business data in bulk.
- Synchronize the data that exists in both Oracle and Siebel databases after the initial load is complete.

Bulk loading of business data is a one-way process. With ODI, business data is:

1. Extracted from the Oracle E-Business Suite (Oracle EBS).
2. Transformed into Siebel database-compatible structure and format.
3. Imported into Siebel Customer Relationship Management (Siebel CRM).

You can also bulk load at a later date to move large amounts of business data, for example, after a business data purchase or an acquisition made by your organization. However, if you have already run ODI and integrated the data to Siebel, then rerunning ODI should not be attempted before performing one of the following:

- Clean up the data that is already synchronized

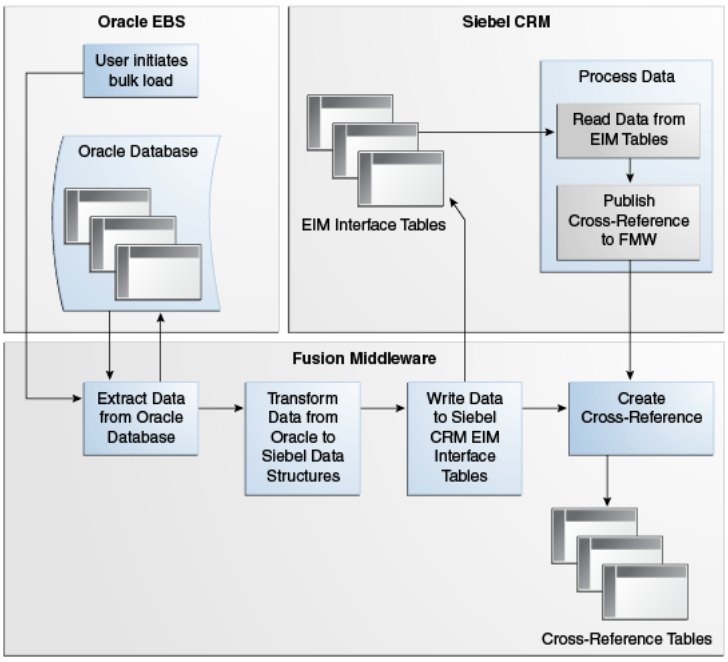
  This option is not recommended as it is both complicated and time-consuming.
- Modify the ODI artifacts to only pull data from the new business unit.

For more information about the steps to load initial data, see Chapter 14, "Running Initial Data Loads."

For more information about using Oracle EBS, ODI, and Enterprise Integration Manager (EIM), see the relevant product documentation.

Figure 2–1 illustrates the bulk load process.

Figure 2–1 Loading Bulk Data from Oracle EBS to Siebel CRM



2.2 Prerequisites

Find prerequisites in Chapter 13, "Reviewing Prerequisites and Data Requirements."

2.3 Solution Assumptions and Constraints

These are the assumptions and constraints for loading initial data.

1. Price lists depend on product data.

   Assets depend on customer and product data. Ensure that you run customer and product loads first.

2. You set up organizations and inventory locations for Siebel CRM and Oracle EBS manually.

3. The DVM data must be synchronized with what the participating applications use.

   You should synchronize this data before running any initial loads or initiating any incremental transactional flows.

4. The fulfillment system (Oracle EBS) may be marking the quantity very high (E+29) just to indicate very high levels, though physically this much quantity would not exist in the inventory.

   It would cause Asset initial load to fail due to ODI being unable to create a temporary table because this value exceeds the allowed limit for the target field in Siebel CRM. Therefore, before importing the data to Siebel CRM, you must set this significantly high number to a number that is more appropriate on the Siebel quantity attribute. In Siebel CRM, the maximum value that the Asset quantity field can hold is E+11.

For more information about organizations and inventory locations, see Chapter 15, "Configuring the Order to Cash Pre-Built Integration."

3

Process Integration for Customer Management

This chapter provides an overview of the process integration for customer management and discusses how to synchronize accounts from Siebel Customer Relationship Management (Siebel CRM) to Oracle E-Business Suite (Oracle EBS). Also discussed are Siebel CRM and Oracle EBS interfaces, core Oracle Application Integration Architecture (Oracle AIA) components, and integration services.

This chapter includes the following sections:

3.1 Process Integration for Customer Management

The process integration for Customer Management between Siebel CRM and Oracle EBS supports the following integration flows:

- **Synchronize new customer accounts from Siebel CRM to Oracle EBS:** This flow enables the synchronization of new customer accounts from Siebel CRM to Oracle EBS as part of booking an order in the process integration for Oracle Order Management.
- **Update and synchronize customer accounts from Siebel CRM to Oracle EBS:** This flow enables the synchronization of customer updates from Siebel CRM to Oracle EBS if the record exists in both Siebel CRM and Oracle EBS. If you update a customer record that does not exist in Oracle EBS, the customer record is not synchronized to Oracle EBS.

- **Synchronize customer accounts from Oracle EBS to Siebel CRM:** In this flow, whenever you create or update a customer account in Oracle EBS, a real-time synchronization flow is initiated to synchronize the related party and customer account to a Siebel customer account.
- **Merge accounts from Oracle EBS to Siebel CRM:** This flow maintains the transactional integrity that is essential to convert orders captured in Siebel CRM into booked orders. Because customer merge events may occur within Oracle EBS, this flow is initiated in Oracle EBS. This flow is initiated when a customer account is merged into another customer account. The merge operation is then performed in Siebel CRM.
- **Merge parties from Oracle EBS to Siebel CRM:** This flow is initiated in Oracle EBS when an organization party is merged into another party. All of the child records that belong to the surviving party in the fulfillment system are synchronized to Siebel CRM after the merge.
- **Bulk load of customer data:** Bulk load enables the extract, transformation, and load of initial customer data from Oracle EBS to Siebel CRM.

  This feature uses Oracle Data Integrator (ODI) to extract relevant customer data from Oracle EBS to load it to the appropriate Siebel CRM Enterprise Integration Manager (EIM) tables and insert it into base tables. This process also enables cross-referencing between Oracle EBS and Siebel CRM.

  For more information about initial data loads, see Chapter 2, "Loading Initial Data."

  For more information about using ODI for bulk processing, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack* "Using Oracle Data Integrator for Bulk Processing".

3.1.1 Prerequisites

Find prerequisites inSection 13.2, "Customer Management: Prerequisites and Data Requirements."

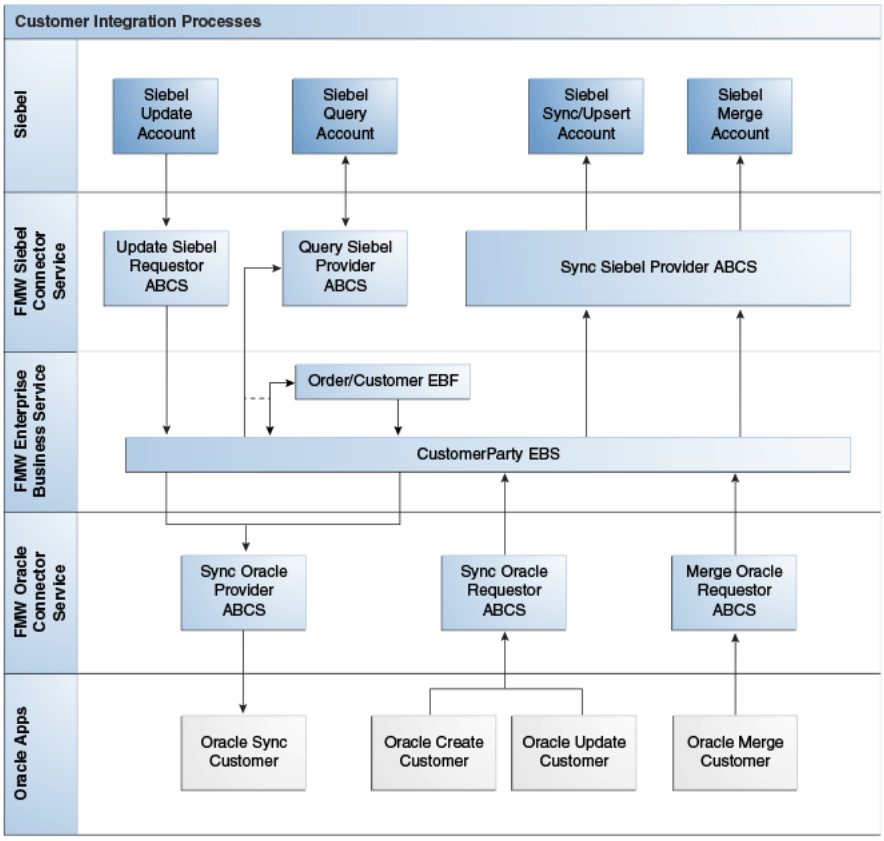
3.1.2 Solution Assumptions and Constraints

The integration design assumes that:

1. The default functionality of the Order to Cash: Siebel CRM - EBS pre-built integration is to only synchronize new customer accounts from Siebel CRM to Oracle EBS when an order is submitted. However, deployments can choose to directly synchronize new accounts from Siebel CRM to Oracle EBS by changing the AIAConfig property `Account.ProcessUpdateEventsOnly` from *True* to *False*.
2. To initiate the Customer synchronization, users must run a concurrent job after any creates or updates to accounts or parties in Oracle EBS. A concurrent job is a batch job that raises events for creates or updates.
3. As part of this integration, the customer number and the site number are auto-generated. In Oracle EBS, the system parameters Auto Generate Customer Number and Auto Generate Site Number are set at the operating unit level. The value for these parameters is set to *Y* so that customer number and site number are autogenerated in Oracle EBS.
4. Account hierarchy synchronization is not in the scope of this release.

Figure 3–1 illustrates the overall flow for the customer process integration.

Figure 3–1 Customer Process Integration Flow



Customer information related to orders is maintained in both Siebel CRM and Oracle EBS, and is synchronized bidirectionally. New accounts can be created and maintained in Siebel CRM. An account is synchronized with Oracle EBS only when an order is booked for it in Siebel CRM.

Whenever a new customer account is created or updated in Oracle EBS, run a concurrent job (TCA Business Object Events - Raise Events) from the Trading Community Architecture (TCA) Manager responsibility. The synchronize flow is initiated after this request finishes successfully. The corresponding primary and bill to or ship to addresses are captured with the account and synchronized. Similarly, the contacts associated with the account are also synchronized.

Any updates to the account (or related addresses or contacts) in Oracle EBS are also synchronized with Siebel CRM. The updates to accounts in Siebel CRM are synchronized to Oracle EBS only if it has been synchronized before.

Whenever party accounts are merged in Oracle EBS, the corresponding accounts in Siebel CRM must be merged or their customers' Ids must be updated accordingly. When an account is inactivated in Oracle EBS, it is inactivated in Siebel CRM.

3.2 Synchronizing New Customer Accounts from Siebel CRM to Oracle EBS

The customer synchronization flow from Siebel CRM to Oracle EBS is orchestrated as part of the Order Process Management flow. This process flow synchronizes new

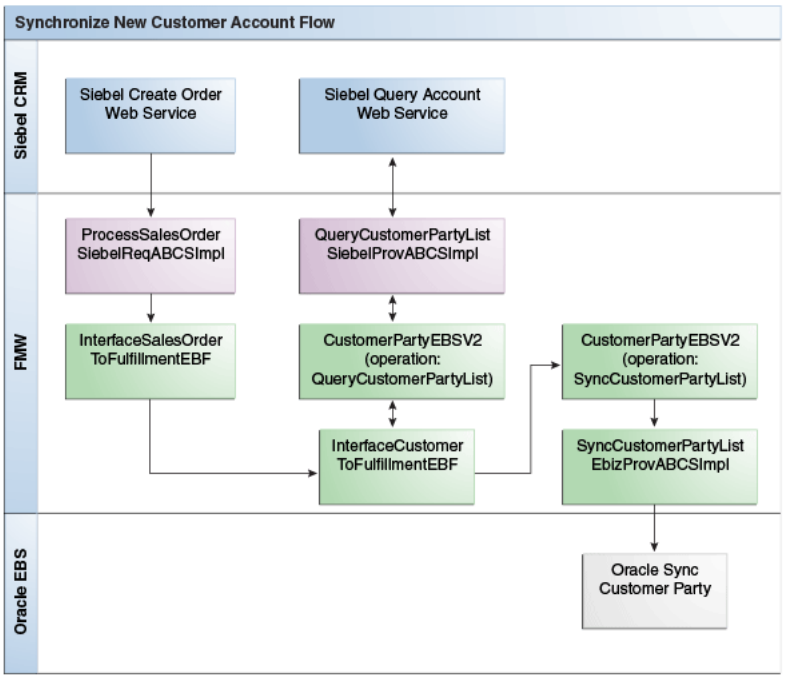
accounts, contacts, and new or existing addressees associated with a Siebel order into Oracle EBS.

When working with the business process for synchronizing new customer accounts from Siebel CRM to Oracle EBS, consider the following guidelines:

- Orders placed on behalf of Siebel accounts result in the creation of an Oracle organization party and a customer account with a corresponding bill-to/ship-to address.
  - Oracle Order application programming interface (API) requires a site Id and customer Id to associate the order to existing records in Oracle EBS; otherwise, a new (or duplicate) customer may be created in Oracle EBS.
  - Account contacts and site contacts may be created in Oracle EBS using the Oracle Order Management interface.
- This account synchronization process creates location, party site, account site, site uses, and contact points in Oracle EBS.

Figure 3–2 illustrates the Synchronize New Customer Account integration flow.

Figure 3–2 Synchronize New Customer Account Integration Flow



3.2.1 Synchronize New Customer Account Integration Flow

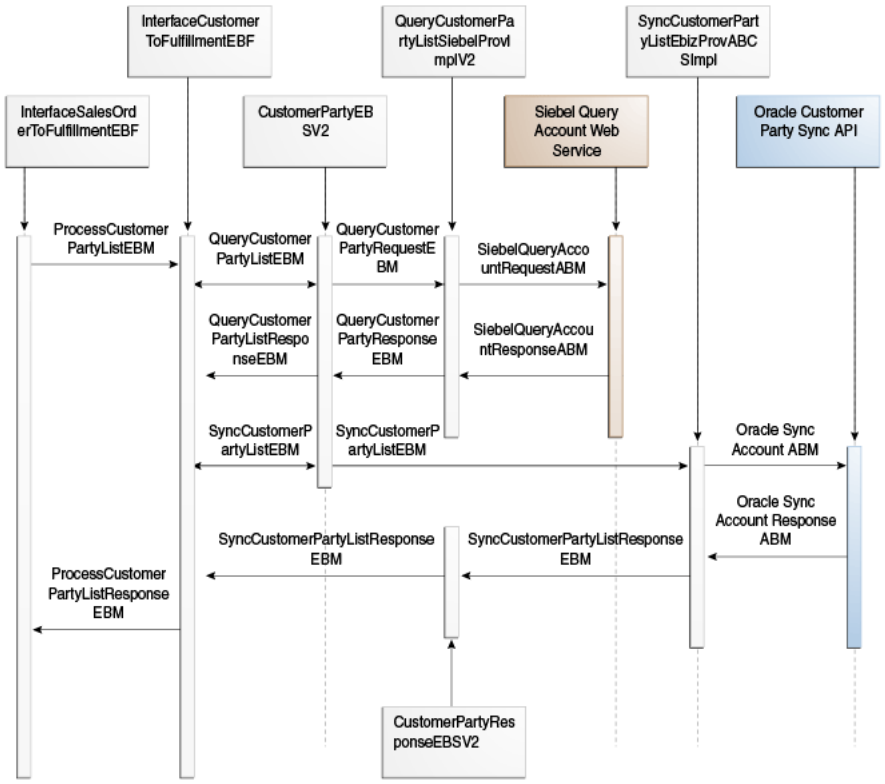
This integration flow uses the following interfaces:

- InterfaceCustomerToFulfillmentEBF
- CustomerPartyEBSV2
- CustomerPartyResponseEBSV2
- QueryCustomerPartyListSiebelProvABCSImplV2
- SyncCustomerPartyListEbizProvABCSImpl

- SyncCustomerPartyListEbizAdapter

Figure 3–3 illustrates the Synchronize Customer Account integration scenario.

Figure 3–3 Synchronize New Customer Account Flow Sequence Diagram



When you initiate the Synchronize New Customer Account process, the following events occur:

1. In Siebel CRM, a user navigates to the Sales Order screen, creates a sales order for the account, changes its status to Booked, and then submits the order.

   These actions trigger the Order Submit process integration, which in turn instantiates the InterfaceCustomerToFulfillmentEBF service.

2. The InterfaceCustomerToFulfillmentEBF invokes the CustomerPartyEBSV2 with the operation QueryCustomerPartyList and the message QueryCustomerPartyListEBM.

   This service fetches account data from Siebel CRM before creating the account in Oracle EBS.

3. Invoking CustomerParyEBSV2 with the operation QueryCustomerPartyList routes the QueryCustomerPartyListEBM to the QueryCustomerPartyListSiebelProvABCSImplV2.

   This service transforms the QueryCustomerPartyListEBM into the Siebel-specific SiebelQueryAccountRequestABM and invokes the Siebel query account web service.

4. The SiebelQueryAccount web service queries the Siebel database and fetches the account data that is sent back as a response using the message SiebelQueryAccountResponseABM.

5. QueryCustomerPartyListSiebelProvABCSImplV2 transforms the response message SiebelQueryAccountResponseABM into the QueryCustomerPartyListResponseEBM.

   This response is sent back to the InterfaceCustomerToFulfillmentEBF through the CustomerPartyEBSV2.

6. The InterfaceCustomerToFulfillmentEBF then invokes the SyncCustomerPartyListEbizProvABCSImpl and sends the account details in the message SyncCustomerPartyListEBM.

   The provider ABC implementation service then transforms the enterprise business message (EBM) into the Oracle EBS-specific message and invokes the Oracle EBS inbound adapter SyncCustomerPartyListEbizAdapter.

7. The Oracle EBS Customer Creation API creates the account in Oracle EBS and returns the account details back in the OracleSyncAccountResponseABM response message.

8. The response from the Oracle EBS API is transformed into the SyncCustomerPartyListResponseEBM, and the cross-reference tables are populated with the new or updated Oracle EBS identifiers.

   The responseEBM is then sent back to the InterfaceCustomerToFulfillmentEBF through the CustomerPartyResponseEBSV2.

9. The response message SyncCustomerPartyListResponseEBM is then transformed into the ProcessCustomerPartyListResponseEBM and sent back to the InterfaceSalesOrderToCustomerEBFV2 using the SalesOrderOrchestrationResponseEBSV2.

   This action signifies the end of the account creation process in the Order Submit integration flow.

3.3 Updating and Synchronizing Customer Accounts from Siebel CRM to Oracle EBS

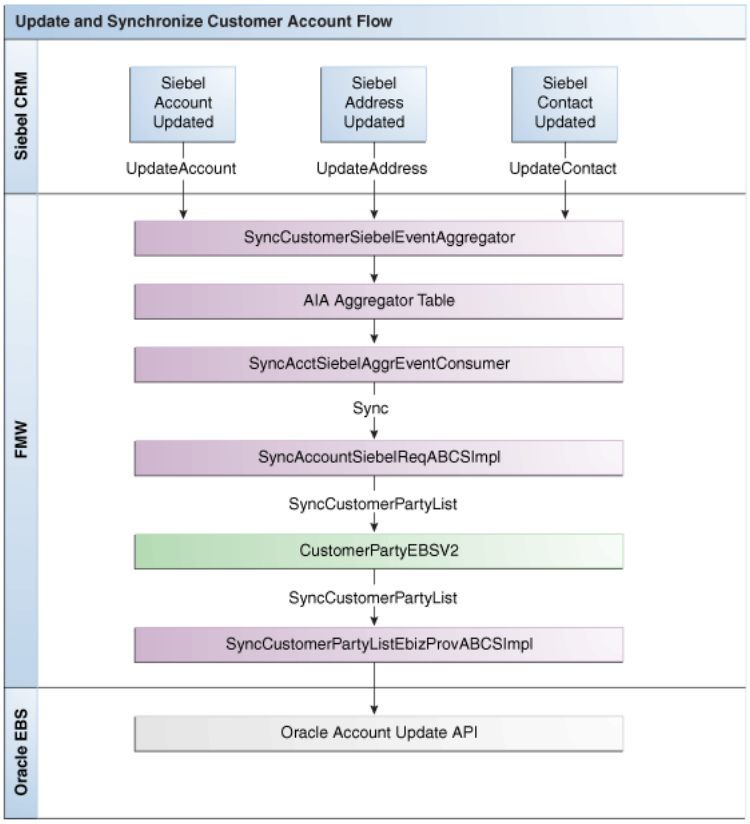
The Update and Synchronize Customer Account integration flow is initiated in Siebel CRM when a customer account is updated.

If a customer account that exists in both systems is updated in Siebel CRM, then that updated customer account must be synchronized to Oracle EBS. If a customer account that does not exist in Oracle EBS is updated, then the customer account is not synchronized to Oracle EBS.

Customer accounts can be updated by making changes to information, such as existing Siebel accounts, and changes to ship-to addresses, bill-to addresses, contact points, account/contact profiles, and so on.

Figure 3–4 illustrates the Update and Synchronize Customer Account integration flow.

Figure 3–4 Update and Synchronize Customer Account Integration Flow



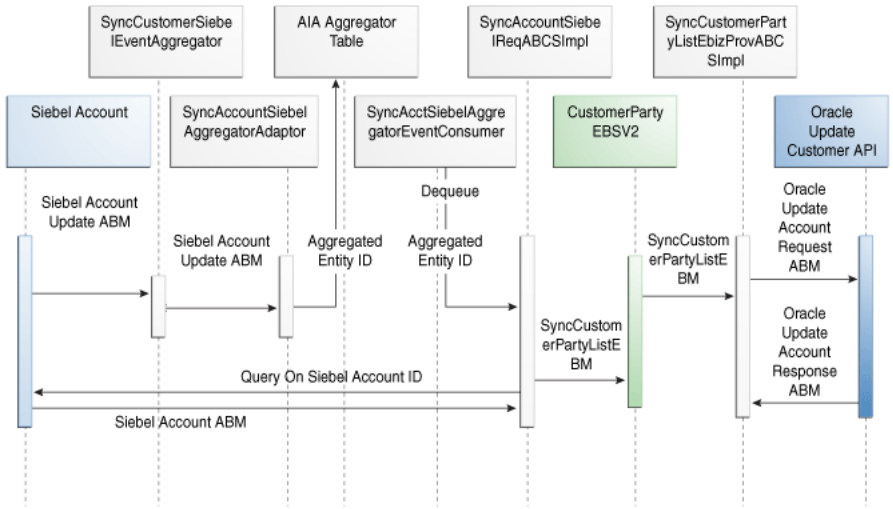
3.3.1 Update and Synchronize Customer Account Integration Flow

This integration flow uses the following interfaces:

- SyncAcctSiebelAggrEventConsumer
- SyncCustomerSiebelEventAggregator
- SyncAccountSiebelReqABCSImpl
- CustomerPartyEBSV2
- SyncCustomerPartyListEbizProvABCSImpl
- SyncCustomerPartyListEbizAdapter

Figure 3–5 illustrates the Update and Synchronize integration scenario.

Figure 3–5 Update and Synchronize Customer Account Flow Sequence Diagram



When you initiate the Update and Synchronize Account process, the following events occur:

1. In Siebel CRM, a user navigates to the Accounts screen, queries an account, and updates an account attribute such as Address or Contact.

   As a result, Siebel CRM invokes the SyncCustomerSiebelEventAggregator with the SiebelAccountUpdatedABM message containing details of the account that was updated.

2. The SyncCustomerSiebelEventAggregator invokes an aggregator adapter (SyncAccountSiebelAggregatorAdapter, SyncAddressSiebelAggregatorAdapter, or SyncContactSiebelAggregatorAdapter) based on the entity updated.

3. The aggregator adapters internally perform the aggregation and enqueue the aggregated entity identifier in the AIA Aggregator table.

4. The message in the table is picked up by the SyncAcctSiebelAggrEventConsumer service and invokes the sync operation of the Siebel requester ABC implementation service.

5. The Siebel Requester ABC implementation, SyncAccountSiebelReqABCSImpl, queries the Siebel application in return with the entity identifier to fetch the entire entity ABM.

   It then transforms the Siebel ABM into the SyncCustomerPartyListEBM and invokes the SyncCustomerPartyList operation of the CustomerPartyEBSV2.

6. Invoking the CustomerPartyEBSV2 with the SyncCustomerPartyList operation routes the SyncCustomerPartyListEBM to the Oracle EBS provider ABC implementation service, SyncCustomerPartyListEbizProvABCSImpl.

7. The Oracle EBS provider ABC implementation service, SyncCustomerPartyListEbizProvABCSImpl, transforms the SyncCustomerPartyListEBM into the appropriate Oracle EBS-specific ABM and invokes the Oracle EBS API using the Oracle EBS adapter SyncCustomerPartyListEbizAdapter.

8. The responses from the Oracle EBS adapter service are transformed into the SyncCustomerPartyListResponseEBM, during which the cross-reference tables are populated with the Oracle EBS identifier values.

The Update and Synchronize Account integration flow ends here.

3.3.1.1 Event Aggregation

Synchronization of accounts updated in Siebel CRM to Oracle EBS is implemented using the Event Aggregation Programming Model.

When an account or its contacts and addresses are updated in Siebel CRM, a fine-grained event is triggered, which invokes the Oracle AIA Aggregator service, `SyncCustomerSiebelEventAggregator`, for every event. This service is a BPEL-based process, which invokes one of three database adapters that reside on the Oracle AIA database. These database adapters expose the Aggregator APIs, which are built using PL/SQL.

In Siebel CRM, an update or create action can lead to multiple events being raised based on the number of entities that were updated or the number of times they were updated. In these cases, The Aggregator is used to aggregate these events and process them in batches instead of processing each event individually.

In this flow, the Event Producer is Siebel CRM, which produces the events that must be aggregated. These messages are routed to the AIA layer by the service `SyncCustomerSiebelEventAggregator`. Based on the message type (Account, Contact or Address entity) the appropriate database adapter is then invoked. The actual aggregation happens in the database, with the data present in the self-referencing table AIA_AGGREGATED_ENTITIES. The PL/SQL APIs contain logic to maintain the relationships between the various entities and to also perform the aggregation based on the parent entity of the event. For this update flow from Siebel CRM to Oracle EBS, the parent entity is usually the Account, which can have one or more contacts and addresses. As part of the aggregation, there is one row per account (parent entity) in the aggregator table. Even if there are multiple events for the same account, there is only one corresponding entry in the aggregator table for the account.

At the end of the PL/SQL processing, the aggregated entities are present in the AIA_AGGREGATED_ENTITIES table. The Aggregator consumer service, `SyncAcctSiebelAggrEventConsumer`, continually polls on this table to initiate the synchronization. The polling interval determines how often messages are polled from the table and can be configured by the user based on the business need. The batch size for the consumer determines how many aggregated entities are picked up for synchronizing in a given transaction and can be changed from the default value. In other words, for a batch size of 1 at a time, only one account is consumed from the aggregator table and synchronized to Oracle EBS. For a batch size of 10, the first 10 accounts in the aggregator table are consolidated into one business message by the consumer service. The consumer service routes the polled entities to the Requester BPEL process, `SyncAccountSiebelReqABCSImpl`. On the Oracle Enterprise Manager Fusion Middleware Control, for every instance of the consumer `SyncAcctSiebelAggrEventConsumer`, there is an instance of the requester service. In scenarios where the batch size is 1, the requester service processes only one account. When the batch size is greater than one, there can be times where multiple accounts are processed in one instance of the `SyncAccountSiebelReqABCSImpl` service.

The message from the consumer routed to the `SyncAccountSiebelReqABCSImpl` service, at this point, contains only the Ids of the entities, like AccountId, AddressId, or ContactId. The complete information of the entity is queried back from Siebel CRM by the Requester service and the EBM is then built based on the retrieved information. The Enterprise Business Service, `CustomerPartyEBSV2`, then routes this EBM (`SyncCustomerPartyListEBM`) to the appropriate Provider service.

The provider service, SyncCustomerPartyListEbizProvABCSImpl, transforms the incoming EBM into the required Oracle EBS specific ABM and invokes the adapter service, SyncCustomerPartyListEbizAdapter. The SyncCustomerPartyListEbizAdapter exposes the Customer API, which creates or updates the account in Oracle EBS. The response from Oracle EBS is used to cross-reference the Ids and complete the transaction.

For more information about the Event Aggregation Programming Model, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Describing the Event Aggregation Programming Model".

3.3.1.2 Updating Polling Interval on FMW for the Aggregator Consumer Services

The PollingInterval for the Account and Contact Aggregator consumer services are, by default, set to 90s and 60s respectively. To change these values for your use cases:

1. Go to the Oracle Enterprise Manager Fusion Middleware Control.
2. Look for the SyncAcctSiebelAggrEventConsumer service located under **Farm_soa_domain, SOA, soa-infra, default.**
3. Select this service and go to the Services and References section.
4. Click service of type *JCA Adapter*.

   Go to the **Properties** tab and set the property **Polling Interval** to the desired value.
5. Click **Apply** to save the changes.
6. Restart the service-oriented architecture (SOA) server for the changes to take effect.

If needed, perform the same steps as for the contact consumer service, SyncContSiebelAggrEventConsumer.

3.4 Synchronizing Customer Accounts from Oracle EBS to Siebel CRM

The Synchronize Customer Account integration flow is initiated in Oracle EBS when a new customer account is created or an existing account is updated. Details relating to the account (account number, addresses, contacts, phone number, fax number, email address, and web address) are reflected in the target system, Siebel CRM.

When an account is created or updated in Oracle EBS, a record appears on a tracking table. Raising a TCA business event causes all of the account records in the tracking table to be stamped with an event identifier and enqueue event message to Oracle Advanced Queuing (AQ). The synchronize account process dequeues from AQ, reads the event identifier, and invokes the TCA API to extract all of the created or updated account information.

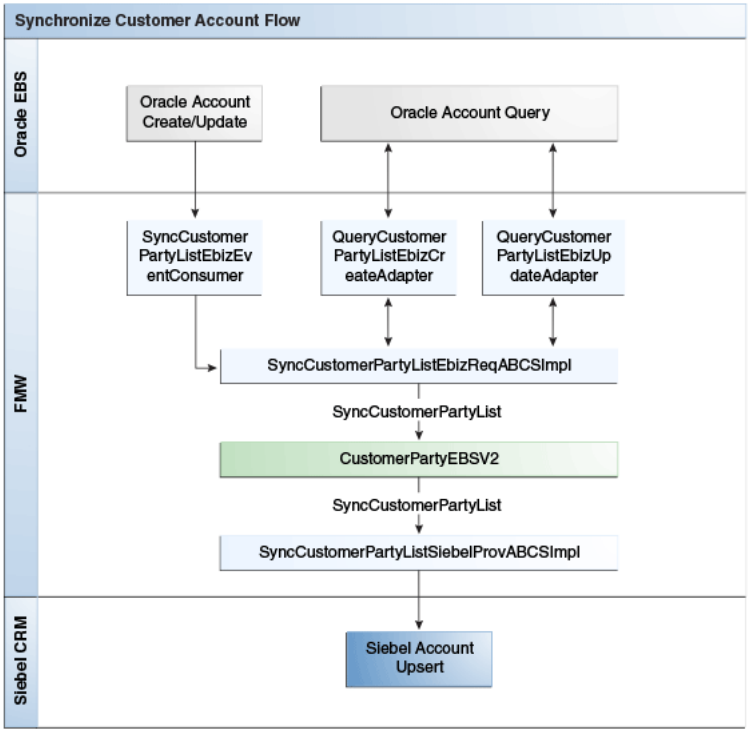
A create or update event is tracked in the following situations:

- When a new party and account is created, the create account business event is raised.
- When a party or account is updated, the update account business event is raised.
- When an account is created on an existing party, the update account business event is raised.

When the SyncCustomerPartyList operation accepts the list of accounts, the requester ABC implementation service passes the entire list of accounts to the CustomerPartyEBSV2 and invokes the corresponding provider ABC implementation service.

Figure 3–6 illustrates the Synchronize Customer Account integration flow.

Figure 3–6 Synchronize Customer Account Integration Flow



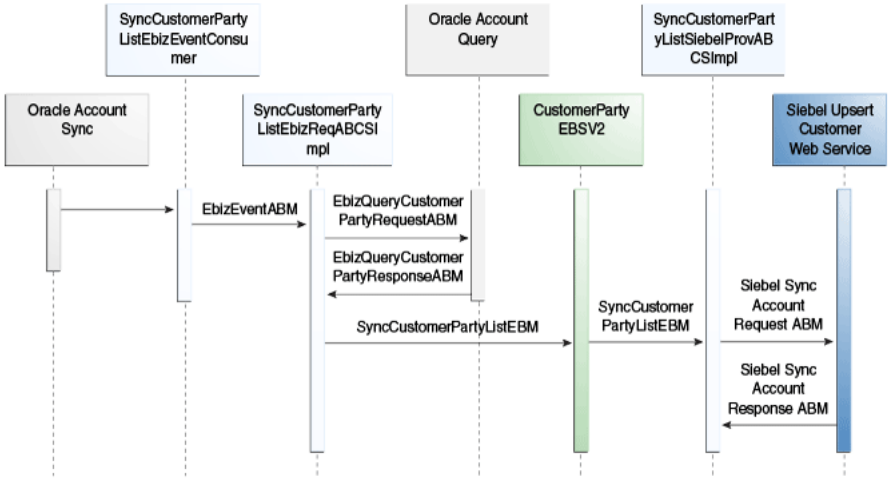
3.4.1 Synchronize Customer Account Integration Flow

This integration flow uses the following interfaces:

- SyncCustomerPartyListEbizEventUpdateConsumer
- SyncCustomerPartyListEbizEventCreateConsumer
- QueryCustomerPartyListEbizCreateAdapter
- QueryCustomerPartyListEbizUpdateAdapter
- SyncCustomerPartyListEbizReqABCSImpl
- CustomerPartyEBSV2
- SyncCustomerPartyListSiebelProvABCSImpl

Figure 3–7 illustrates the Synchronize Customer Account integration scenario.

Figure 3–7 Synchronize Customer Account Flow Sequence Diagram



When you initiate the Synchronize Account process, the following events occur:

1. In Oracle Applications, a user navigates to the Customer Standard form and creates an account with address and contact details.

   Then the user runs the TCA Raise Business Events concurrent program to raise the business events for the account creation. The events are captured in a database table to which the SyncCustomerPartyListEbizEventUpdateConsumer/SyncCustomerPartyListEbizEventCreateConsumer is listening. Different events are raised for a create scenario and an update scenario.

2. The SyncCustomerPartyListEbizEventUpdateConsumer/ SyncCustomerPartyListEbizEventCreateConsumer dequeues the event payload from the queue and routes the request to the SyncCustomerPartyListEbizReqABCSImpl service.

   This service has two operations, create and update, for actions of the same name. The routing occurs based on the payload received from the event.

3. The event payload received by the SyncCustomerPartyListEbizReqABCSImpl contains only the event identifier, not the entire account detail.

   Therefore, account details must be fetched from Oracle EBS based on the Event ID. Using the Query Apps adapters, QueryCustomerPartyListEbizCreateAdapter and QueryCustomerPartyListEbizUpdateAdapter, the account details are fetched based on the kind of action, update, or create.

4. The Query adapters (QueryCustomerPartyListEbizCreateAdapter and QueryCustomerPartyListEbizUpdateAdapter) query the Oracle EBS database with the EbizQueryCustomerPartyRequestABM and fetch the account payload corresponding to the Event ID, which is the EbizQueryCustomerPartyResponseABM.

5. In the SyncCustomerPartyListEbizReqABCSImpl, the QueryResponseABM is transformed into the SyncCustomerPartyListEBM, and the SyncCustomerPartyList operation of the CustomerPartyEBSV2 is invoked with this EBM.

6. The CustomerPartyEBSV2 service routes the SyncCustomerPartyListEBM to the SyncCustomerPartyListSiebelProvABCSImpl.

In this service, the SyncCustomerPartyListEBM is transformed into the SiebelAccountUpsertRequestABM, and the Siebel web service for account creation or update is invoked.

7. The Siebel Customer Update web service does the actual creation or update of the account, address, and contact in Siebel and returns the appropriate response, SiebelAccountUpsertResponseABM.

   This response is received by the SyncCustomerPartyListSiebelProvABCSImpl service.

8. 8.In the SyncCustomerPartyListSiebelProvABCSImpl service, the SiebelAccountUpsertResponseABM is transformed into the SyncCustomerPartyListResponseEBM, during which the cross-reference data is updated with the Siebel identifiers.

   The Account Update flow ends at this point; no response is returned to Oracle EBS.

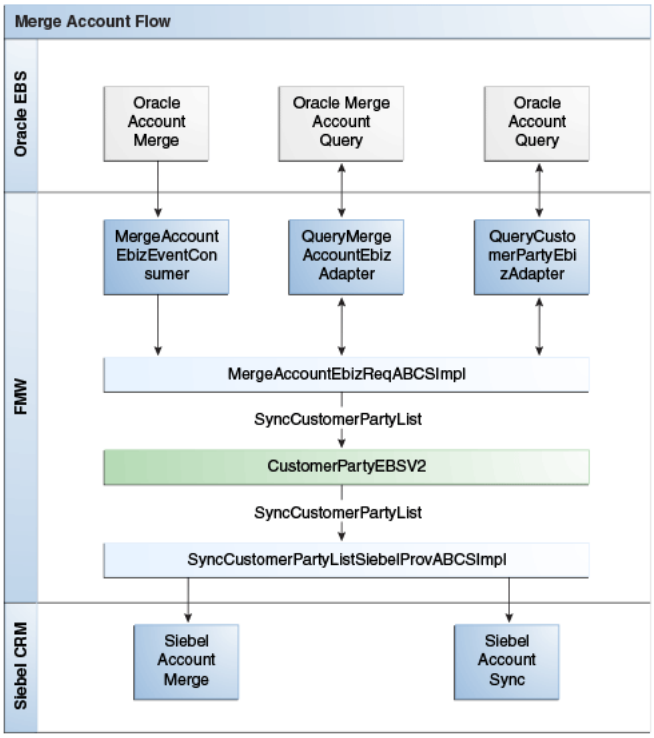
3.5 Merging Accounts from Oracle EBS to Siebel CRM

The Merge Account integration flow is initiated in Oracle EBS when an account is merged into another account in Oracle EBS. A corresponding account merge operation is performed in the target Siebel CRM system.

When accounts are merged in Oracle EBS, a business event is raised in Oracle EBS. The event message contains a customer merge header identifier. The message is enqueued to AQ. The `MergeAccountEbizEventAdapter` dequeues the message and passes it to the `MergeAccountEbizReqABCSImpl`. The process reads the identifier and invokes the TCA API to extract account merge information.

Figure 3–8 illustrates the Merge Account integration flow.

Figure 3–8 Merge Account Integration Flow



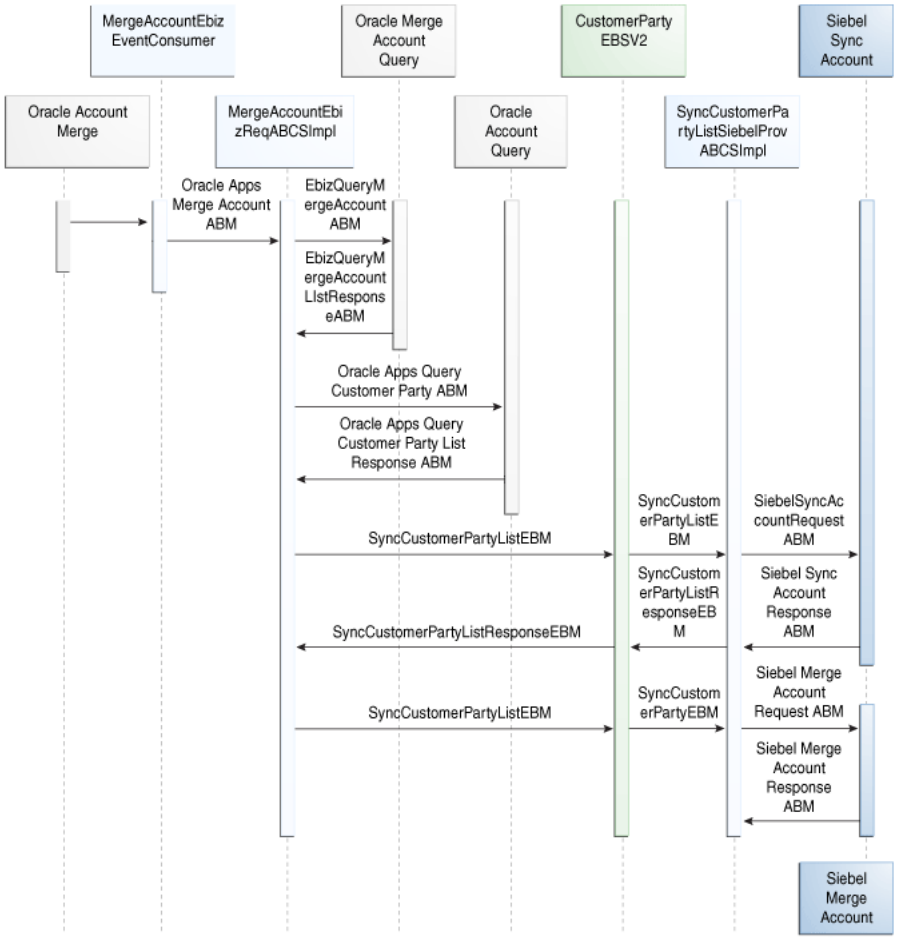
3.5.1 Merge Account Integration Flow

This integration flow uses the following interfaces:

- MergeAccountEbizEventConsumer
- MergeAccountEbizReqABCSImpl
- CustomerPartyEBSV2
- CustomerPartyResponseEBSV2
- SyncCustomerPartySiebelProvABCSImpl
- QueryMergeAccountEbizAdapter
- QueryCustomerPartyEbizAdapter

Figure 3–9 illustrates the Merge Account integration scenario.

Figure 3–9 Merge Account Flow Sequence Diagram



When you initiate the create account process, the following events occur:

1. In Oracle Applications, a user performs an account merge using the Account Merge form.

   The merge involves two entities, the Winning Account (To account) and the Losing Account (From account). The Account Merge operation generates a concurrent request that does the actual merge in Oracle EBS; then it raises the business events to the queue, where a consumer Oracle Apps adapter (MergeAccountEbizEventConsumer) is listening.

2. The MergeAccountEbizEventConsumer dequeues the event payload from the queue and routes the request to the SyncCustomerParty operation of the MergeAccountEbizReqABCSImpl service.

3. The event payload received by the MergeAccountEbizReqABCSImpl contains only the event identifier, not details of the entire account.

   Therefore, the account details must be fetched from Oracle EBS based on the Event ID. You use the Query Apps adapters, QueryMergeAccountEbizAdapter and QueryCustomerPartyEbizAdapter, to fetch the account details.

4. The QueryMergeAccountEbizAdapter fetches the Winning and Losing Account identifiers from Oracle EBS based on the Event ID in the request payload.

This response is then used to build the QueryCustomerPartyABM message, which is then used to fetch the individual account details.

5. The QueryCustomerPartyEbizAdapter service fetches the account details (addresses, contact points, contacts, and so on) from Oracle EBS and sends the response back as QueryAppsCustomerPartyListResponseABM.
6. The response from the QueryCustomerPartyEbizAdapter is transformed into the SyncCustomerPartyListEBM by the MergeAccountEbizReqABCSImpl service.

   Then the SyncCustomerPartyList operation of the CustomerPartyEBSV2 service is invoked.
7. The CustomerPartyEBSV2 service routes the SyncCustomerPartyListEBM to the SyncCustomerPartySiebelProvABCSImpl, which then transforms the EBM to the Siebel SyncAccountRequestABM and invokes the Sync Siebel web service.

   The response sent by the Sync Siebel web service is then transformed into the SyncCustomerPartyListResponseEBM, during which the cross-reference data is updated. The response is sent back to the MergeAccountEbizReqABCSImpl using the CustomerPartyResponseEBSV2 service.
8. The Account Merge into Siebel occurs in two stages

   First, the Losing account data is synchronized to the Winning account (based on an option that, when selected during the Merge, creates the same address using the Sync Siebel web service (until step 7). Second, the Merge Siebel web service is used to perform the actual merge in Siebel.
9. As the second part of the merge, the MergeAccountEbizReqABCSImpl invokes the SyncCustomerPartyList operation of the CustomerPartyEBSV2 service, which in turn routes the EBM to the SyncCustomerPartyListSiebelProvABCSImpl.
10. Based on a flag in the EBM, the SyncCustomerPartyListSiebelProvABCSImpl transforms the SyncCustomerPartyListEBM to the SiebelMergeAccountReqABM and invokes the Merge Siebel web service to perform the actual merge in the Siebel application.

    The response received from the web service is then used to update the cross-reference data while it is being transformed to the SyncCustomerPartyListResponseEBM.
11. The Merge Account flow ends with no further responses returned to Oracle EBS.

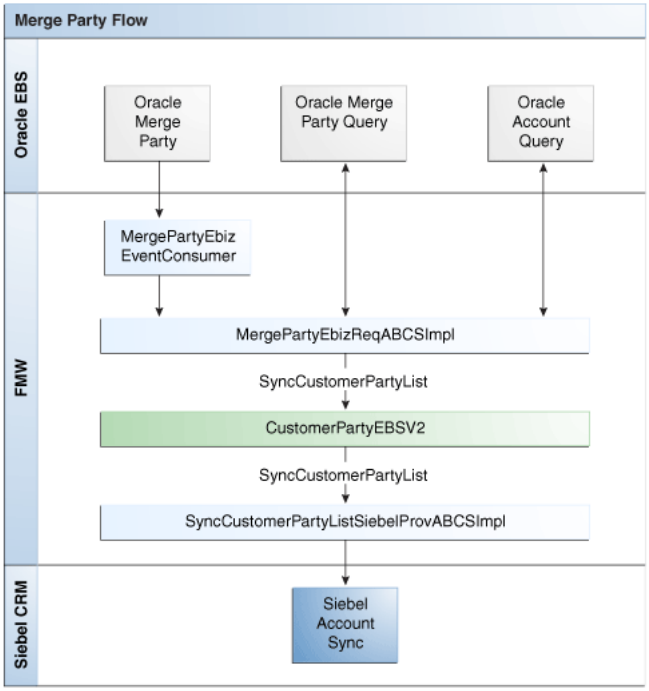
3.6 Merging Parties from Oracle EBS to Siebel CRM

The Merge Party integration flow is initiated in Oracle EBS when an organization party is merged into another party in Oracle EBS. Because the Party Id is part of the Oracle Id in the Id cross-reference, the Merge Party flow updates the corresponding Oracle Id value in the Id cross-reference. In addition, all accounts of the surviving party in Oracle EBS are synchronized to Siebel CRM after the merge.

When parties are merged in Oracle EBS, a business event is raised in Oracle EBS. The event message contains the batch and winning party identifier. The message is enqueued to AQ. The `MergePartyEbizEventAdapter` dequeues the message from AQ and passes it to the `MergePartyEbizReqABCSImpl`. The process reads the identifiers and invokes the TCA API to extract merge party information.

Figure 3–10 illustrates the Merge Party integration flow.

Figure 3–10 Merge Party Integration Flow



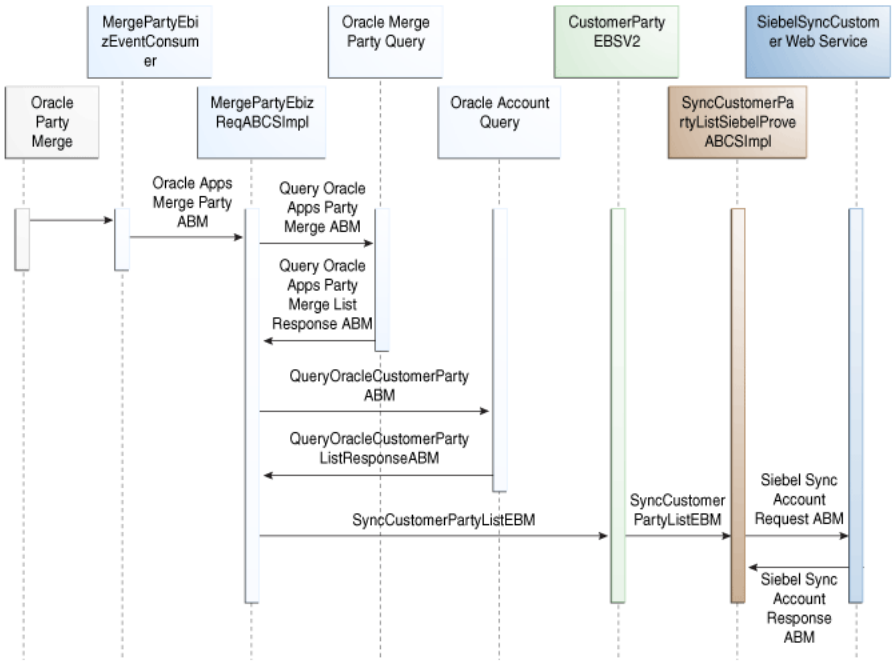
3.6.1 Merge Party Integration Flow

This integration flow uses the following interfaces:

- MergePartyEbizEventConsumer
- MergePartyEbizReqABCSImpl
- CustomerPartyEBSV2
- SyncCustomerPartySiebelProvABCSImpl
- QueryRelatedOrgCustEbizAdapter
- QueryMergeOrgCustEbizAdapter
- QueryPartyMergeEbizAdapter

Figure 3–11 illustrates the Merge Party integration scenario.

Figure 3–11 Merge Party Flow Sequence Diagram



When you initiate the Merge Party process, the following events occur:

1. In Oracle Applications, a user performs a party merge using the Party Merge form. The merge involves two entities, the Winning Party (To account) and the Losing Party (From account).

   The Party Merge operation generates a concurrent request that does the actual merge in Oracle EBS and then raises the business events to the queue, where a consumer Oracle EBS adapter (MergePartyEbizEventConsumer) is listening.

2. The MergePartyEbizEventConsumer dequeues the event payload from the queue and routes the request to the SyncCustomerParty operation of the MergePartyEbizReqABCSImpl service.

3. The event payload received by the MergePartyEbizReqABCSImpl contains only the event identifier, not details of the entire account.

   Therefore, account details must be fetched from Oracle EBS based on the Event ID. Use the Query Apps adapters, QueryPartyMergeEbizAdapter, QueryMergeOrgCustEbizAdapter, and QueryRelatedOrgCustEbizAdapter, to fetch the account details. When the parties being merged are of type Organization, use QueryMergeOrgCustEbizAdapter. When the parties being merged are not of type Organization, use QueryRelatedOrgCustEbizAdapter.

4. The QueryPartyMergeEbizAdapter fetches the Winning and Losing Account identifiers from Oracle EBS based on the event ID in the request payload.

   This response is then used to build the QueryCustomerPartyABM message, which is then used to fetch the individual account details.

5. The QueryMergeOrgCustEbizAdapter service fetches account details for the Organization type parties (addresses, contact points, contacts, and so on) from Oracle EBS and sends the response back as QueryAppsCustomerPartyListResponseABM.

The QueryRelatedOrgCustEbizAdapter service fetches account details for parties being merged that are not of type Organization.

6. The response from the QueryMergeOrgCustEbizAdapter (or QueryRelatedOrgCustEbizAdapter) is transformed into the SyncCustomerPartyListEBM by the MergePartyEbizReqABCSImpl service, which then invokes the SyncCustomerPartyList operation of the CustomerPartyEBSV2 service.
7. The CustomerPartyEBSV2 service routes the SyncCustomerPartyListEBM to the SyncCustomerPartySiebelProvABCSImpl, which then transforms the EBM into the Siebel SyncAccountRequestABM and invokes the Sync Siebel web service.

   Because no parties are available in Siebel, the Merge service is not used in this flow. Instead, the Sync service is used to synchronize the accounts of the parties being merged in Oracle EBS.
8. The response sent by the Sync Siebel web service is then transformed into the SyncCutomerPartyListResponseEBM, during which the cross-reference data is updated.

   The Merge Party flow ends at this point, and no further responses are returned to Oracle EBS.

3.7 Siebel CRM Interfaces

These Siebel CRM web services are available for the Customer Management integration flow:

Inbound Siebel CRM Web Services

- Service Name: SWICustomerParty
  - Operation Name: SWICustomerPartyInsertOrUpdate
  - Request Schema: SWICustomerPartyInsertOrUpdate_Input
  - Response Schema: SWICustomerPartyInsertOrUpdate_Output
- Service Name: SWICustomerParty
  - Operation Name: SWICustomerPartyQueryByExample
  - Request Schema: SWICustomerPartyQueryByExample_Input
  - Response Schema: SWICustomerPartyQueryByExample_Output
- Service Name: SWICustomerParty
  - Operation Name: SWICustomerPartySynchronize
  - Request Schema: SWICustomerPartySynchronize_Input
  - Response Schema: SWICustomerPartySynchronize_Output
- Service Name: SWICustomerParty
  - Operation Name: SWIMergeServices
  - Request: SchemaSWIMergeServicesMerge_Input
  - Response Schema: SWIMergeServicesMerge_Output

Outbound Siebel CRM Web Services

These events are raised in Siebel and are consumed by this integration:

- Account Updated
  - Siebel calls the SyncCustomerSiebelEventAggregator service with an AccountUpdatedABM.
- Contact Updated
  - Siebel calls the SyncCustomerSiebelEventAggregator service with a ContactUpdatedABM.
- Address Updated
  - Siebel calls the SyncCustomerSiebelEventAggregator service with an AddressUpdatedABM.

These are the Siebel outbound services:

- CalculateShippingChargeSalesOrderSiebelReqABCSImpl
- CheckATPSalesOrderSiebelReqABCSImpl
- ConfiguratorUserLangSiebelAdapter
- CreditCheckSalesOrderSiebelReqABCSImpl
- FetchAccountSiebelReqABCSImpl
- FetchContactSiebelReqABCSImpl
- MatchAccountSiebelReqABCSImpl
- MatchContactSiebelReqABCSImpl
- PaymentAuthorizationSalesOrderSiebelReqABCSImpl
- ProcessQuoteSiebelJMSProducer
- ProcessSalesOrderSiebelJMSProducerV2
- RequestProductStructureSiebelJMSProducer
- SyncCustomerSiebelEventAggregator
- ESB_ConfiguratorCopyConfigEbizAdapter_Service

For more information about Siebel web services, see *CRM Web Services Reference*.

3.8 Oracle EBS Interfaces

These Oracle EBS web services are available for the Customer Management flow integration:

Inbound to Oracle EBS Web Services

This integration uses these Oracle artifacts:

- Service Name: HZ_ORG_CUST_BO_PUB.GET_ORG_CUSTS_CREATED
- Service Name: HZ_ORG_CUST_BO_PUB.GET_ORG_CUSTS_UPDATED
- Service Name: HZ_AIA_CUSTOM_PKG. SYNC_ACCT_ORDER
- Service Name: HZ_AIA_CUSTOM_PKG.get_acct_merge_obj
- Service Name: HZ_AIA_CUSTOM_PKG.get_party_merge_objs
- Service Name: HZ_AIA_CUSTOM_PKG.get_merge_org_custs
- Service Name: HZ_AIA_CUSTOM_PKG.get_related_org_cust_objs

Outbound from Oracle EBS Event Interfaces

These events are raised in Oracle and are consumed by this integration:

- oracle.apps.ar.hz.OrgCustBO.create
- oracle.apps.ar.hz.OrgCustBO.update
- oracle.apps.ar.hz.CustAccount.merge
- oracle.apps.ar.hz.Party.merge

For more information about Oracle EBS web services and documentation prior to Release 12.1.3, see the library on Oracle Technology Network: http://www.oracle.com/technetwork/documentation/applications-167706.html?. For Oracle EBS documentation for R12.1.3 and beyond, see this library: http://download.oracle.com/docs/cd/E18727_01/index.htm?

3.9 Core Oracle AIA Components

The Customer Management integration uses the following horizontal components:

- CustomerPartyEBO
- InterfaceCustomerToFulfillmentEBF
- QueryCustomerPartyListEBM
- QueryCustomerPartyListResponseEBM
- SyncCustomerPartyListEBM
- SyncCustomerPartyListResponseEBM
- CustomerPartyEBS

The industry enterprise business object (EBO) and enterprise business message XML schema (EBM XSD) files are located here: $AIA_HOME/AIAMetaData/AIAComponents/EnterpriseObjectLibrary/Core/EBO/

The industry enterprise business service (EBS) WSDL files are located here: $AIA_HOME/AIAMetaData/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/

For detailed documentation of individual EBOs and EBMs, click the AIA Reference Doc link on EBO and EBM detail pages in Oracle Enterprise Repository.

For more information about using the Oracle Enterprise Repository and configuring it to provide the AIA Reference Doc link, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring and Using Oracle Enterprise Repository as the Oracle AIA SOA Repository."

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

For more information, see *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Extensibility."

3.10 Integration Services

These services are delivered with this integration:

- CustomerPartyEBSV2

- CustomerPartyResponseEBSV2
- InterfaceCustomerToFulfillmentEBF
- SyncAccountSiebelReqABCSImpl
- QueryCustomerPartyListSiebelProvABCSImplV2
- SyncCustomerPartyListSiebelProvABCSImpl
- SyncCustomerPartyListEbizReqABCSImpl
- SyncCustomerPartyListEbizProvABCSImpl
- MergeAccountEbizReqABCSImpl
- MergePartyEbizReqABCSImpl
- SyncCustomerSiebelEventAggregator
- SyncAccountSiebelAggregatorAdapter
- SyncContactSiebelAggregatorAdapter
- SyncAddressSiebelAggregatorAdapter
- SyncAcctSiebelAggrEventConsumer
- SyncContSiebelAggrEventConsumer
- SyncCustomerPartyListEbizEventUpdateConsumer
- SyncCustomerPartyListEbizEventCreateConsumer
- MergeAccountEbizEventConsumer
- MergePartyEbizEventConsumer
- SyncCustomerPartyListEbizAdapter

For more information, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring and Using Oracle Enterprise Repository as the Oracle AIA SOA Repository."

3.10.1 CustomerPartyEBSV2

The CustomerPartyEBSV2 Enterprise Business Service is routing service that exposes all of the enterprise operations that can be performed with a CustomerParty enterprise object. All of the Customer Management integration flows make use of the operations provided by this enterprise business service.

The CustomerPartyEBSV2 service uses these operations:

- QueryCustomerPartyList
- SyncCustomerPartyList

For more information about this EBS, see the *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Designing and Developing Enterprise Business Services" and *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Enterprise Business Services."

3.10.2 CustomerPartyResponseEBSV2

The CustomerPartyResponseEBSV2 Enterprise Business Service is a routing service that exposes all of the enterprise response operations that can be performed with a

CustomerParty enterprise object. All of the Customer Management integration flows make use of the response operations provided by this.

The CustomerPartyResponseEBSV2 EBS uses this operation SyncCustomerPartyListResponse.

For more information about this EBS, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Designing and Developing Enterprise Business Services" and *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Enterprise Business Services."

3.10.3 InterfaceCustomerToFulfillmentEBF

The InterfaceCustomerToFulfillmentEBF enterprise business flow is implemented as an asynchronous business process execution language (BPEL) process, using the asynchronous message exchange pattern (with delayed response) framework of Oracle AIA.

This enterprise business flow creates or synchronizes all the customer accounts and related components (such as address and contacts) in sequence to the appropriate fulfillment system. This service is invoked from the Order integration flow with a list of Customer Account IDs, Address IDs and contact IDs, and the Target System ID. When the process is complete, a response is returned to the Order flow confirming that all accounts, addresses, and contacts have been set up in the target fulfillment system and that Order processing can continue. In case of an error, an error code is returned, signaling that Order processing cannot continue.

Assumption/Constraint: A single target back-office system is identified within the Order flow, and this information is passed to this service through the EBM header. The source CRM system must also pass through the EBM header.

This service involves a single operation, and the input is an instance of the ProcessCustomerPartyListEBM message containing the target fulfillment system identifier in the EBM header. The Data Area of the message contains one or more customer account IDs and, for each account, one or more address and contact IDs that must be synchronized to the target fulfillment system. Also, enough identifying information must be available to support querying back to the proper CRM source system for the detailed account, customer, and contact data.

The service looks up the cross-reference values for Organization ID. If any of the IDs are not present in the source CRM system, the processes ends in error.

The service also establishes the cross-reference for Account ID, Contact ID, and Address ID between Common Values and corresponding Oracle Apps IDs.

The return message contains the same list of account IDs that were passed in the request, with additional flags to indicate success or failure and appropriate error messages, for each account.

The service also assumes that before calling the Order process, invoking this service would have established the cross-reference for the Siebel to Common ID for all new accounts, addresses, and contacts that are part of the Siebel sales order or quote.

For more information about this flow, see *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding the Oracle AIA Reference Architecture," AIA Service Artifacts.

3.10.4 SyncAccountSiebelReqABCSImpl

The SyncAccountSiebelReqABCSImpl service is responsible for transforming the Siebel messages into the appropriate Sync Customer Account EBM format and invoking the SyncCustomerPartyList operation of the CustomerPartyEBSV2.

3.10.5 QueryCustomerPartyListSiebelProvABCSImplV2

The QueryCustomerPartyListSiebelProvABCSImplV2 service is invoked by the CustomerPartyEBSV2 when the routing rules determine that Siebel is to be the service provider for the CustomerPartyEBS-QUERY operation. This determination occurs during the integration flow that is initiated by the account sync portion of the order flow.

This service has one synchronous request/reply operation: QueryCustomerPartyList.

The input and output messages are instances of the QueryCustomerPartyListRequestEBM and QueryCustomerPartyListResponseEBM, respectively.

3.10.6 SyncCustomerPartyListSiebelProvABCSImpl

CustomerPartyEBSV2 invokes the SyncCustomerPartyListSiebelProvABCSImpl service when the routing rules determine that Siebel is to be the service provider for the SyncCustomerPartyList EBS operation. This occurs in the following scenarios:

When you create or update a customer account in Oracle Apps. This service is initiated to synchronize customer accounts from Oracle to Siebel. An update web service call to Siebel is invoked.

Before you merge an account, this service is initiated to synchronize surviving account information from Oracle to Siebel. In the sync operation, if an address with row ID 100 is in Siebel but not in the request message, then the address is removed. A synchronize web service call to Siebel is invoked.

This service has one synchronous request/reply operation: SyncCustomerParty.

The input and output messages are instances of the SyncCustomerPartyListEBM and SyncCustomerPartyListResponseEBM, respectively.

3.10.7 SyncCustomerPartyListEbizReqABCSImpl

The SyncCustomerPartyListEbizReqABCSImpl service is responsible for transforming the Oracle EBS account, contact, or address create or update event message into the appropriate SyncCustomerPartyListEBM format and invoking the SyncCustomerPartyList operation of the CustomerPartyEBSV2.

This service has two asynchronous operations: create and update.

3.10.8 SyncCustomerPartyListEbizProvABCSImpl

The CustomerPartyEBSV2 invokes the SyncCustomerPartyListEbizProvABCSImpl service when the routing rules determine that Oracle Apps is to be the service provider of the SyncCustomerPartyList EBS operation. This determination occurs during the integration flow that is initiated by the account create processing portion of the Sales Order flow.

This service has one synchronous request/reply operation: SyncCustomerPartyList.

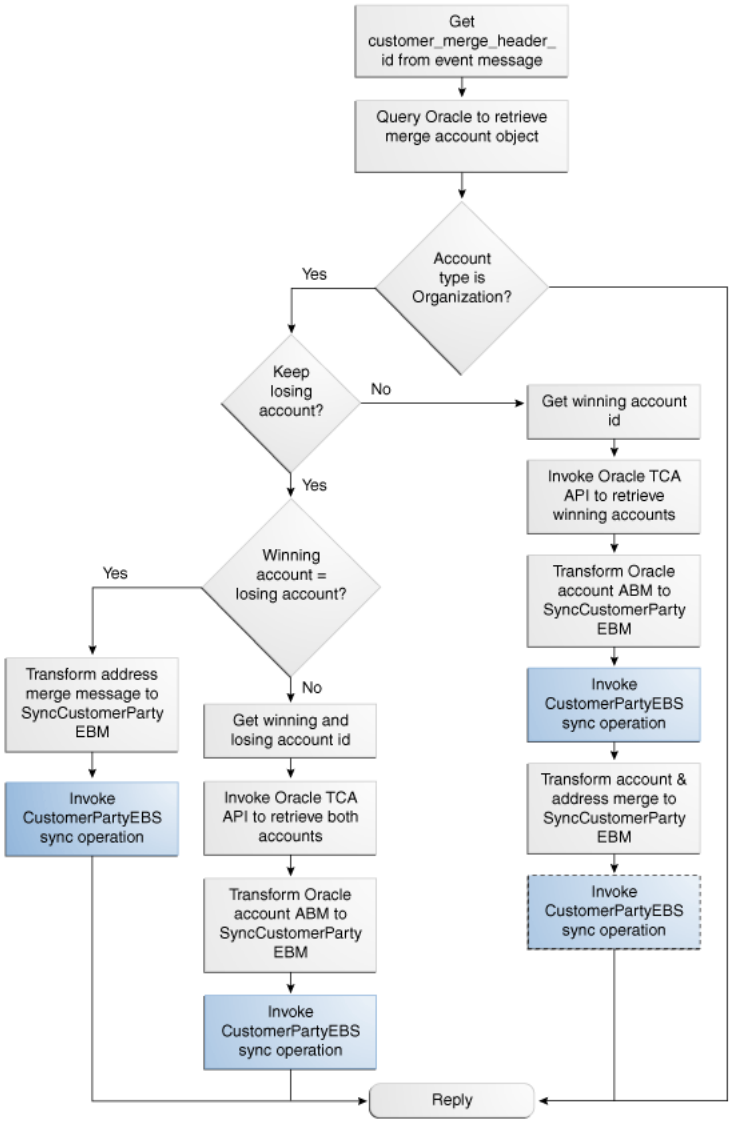
3.10.9 MergeAccountEbizReqABCSImpl

The MergeAccountEbizReqABCSImpl service is responsible for transforming the Oracle Apps account, contact, or address merge event message into the appropriate SyncCustomerPartyListEBM format and invoking the CustomerPartyEBSV2.

This service has one asynchronous request operation: SyncCustomerParty.

Figure 3–12 illustrates the MergeAccountEbizReqABCSImpl process activity:

Figure 3–12 MergeAccountEbizReqABCSImpl Activity Diagram



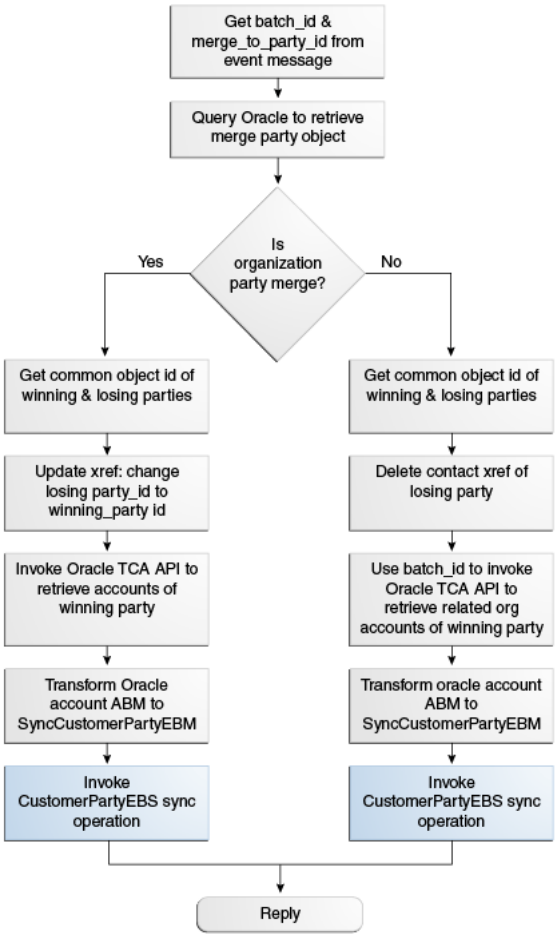
3.10.10 MergePartyEbizReqABCSImpl

The MergePartyEbizReqABCSImpl service is responsible for transforming the Oracle Apps account, contact, or address merge event message into the appropriate SyncCustomerPartyEBM format.

This service has one asynchronous request/reply operation: SyncCustomerParty.

Figure 3–13 illustrates the MergePartyEbizReqABCSImpl process activity:

Figure 3–13 MergePartyEbizReqABCSImpl Activity Diagram



3.10.11 SyncCustomerSiebelEventAggregator

The SyncCustomerSiebelEventAggregator is implemented as a BPEL process because it involves Java Message Service (JMS) header manipulations. This service has three operations, one for each kind of event raised by the Siebel application when an account, contact, or address is created or updated. Each operation invokes the corresponding aggregator adapter service, which does the aggregation of the event in the AIA Aggregator table. The following operations are defined in this service:

- aggregateaccountevent
- aggregatecontactevent
- aggregateaddressevent

For more information about the event aggregation design pattern, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Describing the Event Aggregation Programming Model."

3.10.12 SyncAccountSiebelAggregatorAdapter

The SyncAccountSiebelAggregatorAdapter is implemented as a process with a database adapter and routing services. This service aggregates the account events generated in the Siebel application when an account is created or updated. This service

invokes a PL/SQL procedure, AIA_AGGREGATOR_PUB.SIEBEL_AGGREGATE_ACCOUNT, which does the actual aggregation in the AIA aggregator table.

For more information about the event aggregation design pattern, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Describing the Event Aggregation Programming Model."

3.10.13 SyncContactSiebelAggregatorAdapter

The SyncContactSiebelAggregatorAdapter is implemented as a process with a database adapter and routing services. This service aggregates the account events generated in the Siebel application when an account is created or updated. This service invokes a PL/SQL procedure, AIA_AGGREGATOR_PUB.SIEBEL_AGGREGATE_CONTACT, which does the actual aggregation in the AIA aggregator table.

For more information about the event aggregation design patter *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Describing the Event Aggregation Programming Model."

3.10.14 SyncAddressSiebelAggregatorAdapter

The SyncAddressSiebelAggregatorAdapter is implemented as a process with a database adapter and routing services. This service aggregates the account events generated in the Siebel application when an account is created or updated. This service invokes a PL/SQL procedure, AIA_AGGREGATOR_PUB.SIEBEL_AGGREGATE_ADDRESS, which does the actual aggregation in the AIA aggregator table.

For more information about the event aggregation design pattern, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Describing the Event Aggregation Programming Model."

3.10.15 SyncAcctSiebelAggrEventConsumer

The SyncAcctSiebelAggrEventConsumer service is responsible for dequeueing the aggregated Siebel Account identifiers from the AIA Aggregator table and invoking the appropriate operation of the Requestor application business connector service (ABCS), SyncAccountSiebelReqABCSImpl.

3.10.16 SyncContSiebelAggrEventConsumer

The SyncContSiebelAggrEventConsumer service is responsible for dequeueing the aggregated Siebel Contact identifiers from the AIA Aggregator table and invoking the appropriate operation of the Requestor ABCS, SyncContactSiebelReqABCSImpl.

3.10.17 SyncCustomerPartyListEbizEventCreateConsumer

The SyncCustomerPartyListEbizEventCreateConsumer service is responsible for dequeueing the Oracle create account business event payloads from AQ and invoking the appropriate operation of the Requestor ABCS, SyncCustomerPartyEbizReqABCSImpl. The dequeue operation is done depending on the CorrelationID in AQ. For customer create, the correlation ID is oracle.apps.ar.hz.OrgCustBO.create.

Two business events, each with one operation, are available for reading each of the three types of Oracle Apps messages from the AQ:

- oracle.apps.ar.hz.OrgCustBO.create.

- oracle.apps.ar.hz.OrgCustBO.update

3.10.18 SyncCustomerPartyListEbizEventUpdateConsumer

The SyncCustomerPartyListEbizEventUpdateConsumer service is responsible for dequeueing the Oracle create or update account business event payloads from AQ and invoking the appropriate operation of the Requestor ABCS, SyncCustomerPartyEbizReqABCSImpl. The dequeue operation is done depending on the CorrelationID in AQ. For customer create, the correlation ID is oracle.apps.ar.hz.OrgCustBO.create. For customer update, the correlation ID is oracle.apps.ar.hz.OrgCustBO.update.

Two business events, each with one operation, are available for reading each of the three types of Oracle Apps messages from the AQ:

- oracle.apps.ar.hz.OrgCustBO.create
- oracle.apps.ar.hz.OrgCustBO.update

3.10.19 MergeAccountEbizEventConsumer

The MergeAccountEbizEventConsumer service is responsible for dequeueing the Oracle account merge business event payloads from AQ and invoking the appropriate operation of the Requestor ABCS. For the account merge, the correlation ID is oracle.apps.ar.hz.CustAccount.merge and the event adapter is MergeAccountEbizEventAdapter. The dequeue operation is done depending on the CorrelationID in AQ.

One business event with one operation is available for reading each of the three types of Oracle Apps messages from the AQ oracle.apps.ar.hz.CustAccount.merge.

3.10.20 MergePartyEbizEventConsumer

The MergePartyEbizEventConsumer service is responsible for dequeueing the Oracle party merge business event payloads from AQ and invoking the appropriate operation of the Requestor ABCS. For the party merge, the correlation ID is oracle.apps.ar.hz.Party.merge and the event adapter is MergePartyEbizEventAdapter. The dequeue operation is done depending on the CorrelationID in AQ.

One business event with one operation is available for reading each of the three types of Oracle Apps messages from the AQ oracle.apps.ar.hz.Party.merge.

3.10.21 SyncCustomerPartyListEbizAdapter

The SyncCustomerPartyListEbizAdapter service is an Oracle EBS Adapter service registered in Mediator. This adapter service exposes the HZ_AIA_CUSTOM_PUB.sync_acct_order PL/SQL API. This wrapper API is delivered as part of EBS HZ.N.

This service is the interface through which an order is created in Oracle EBS. SyncCustomerPartyListEbizProvABCSImpl invokes this service as part of the Update Customer and Create Order integration flow.

The SyncCustomerPartyListEbizAdapter service exposes the Sync customer party operation of the PL/SQL wrapper API. This operation is a synchronous request + reply operation. If you register this adapter service in Mediator, it exposes a SOAP binding, which is used in this integration to invoke the service from the SyncCustomerPartyListEbizProvABCSImpl.

4

Process Integration for Product Management

This chapter provides an overview of the process integration for Product Management and discusses Item and Bill of Material (BoM) synchronization, Siebel Customer Relationship Management (Siebel CRM) and Oracle E-Business Suite (Oracle EBS) interfaces, core Oracle Application Integration Architecture (Oracle AIA) components, and integration services.

This chapter includes the following sections:

4.1 Process Integration for Product Management

The process integration for product management supports the following integration flows:

- **Synchronize Items:** This flow enables the synchronization of items from Oracle EBS to simple products in Siebel CRM. This one-way feed from Oracle EBS to Siebel CRM is initiated by Oracle EBS to create or update an item from Oracle EBS to a simple product in Siebel CRM.
- **Synchronize BoMs:** This flow enables the synchronization of BoM structures from Oracle EBS to complex products in Siebel CRM. This process replicates a new or updated BoM from Oracle EBS to Siebel CRM as a configurable or customizable product. This one-way feed from Oracle EBS to Siebel CRM is initiated by Siebel CRM to create or update a configurable or customizable product in Siebel CRM.
- **Initial load of Items:** This flow enables the extract, transformation, and load (ETL) of items from Oracle EBS to Siebel CRM. This feature uses Oracle Data Integrator (ODI) to extract relevant item information from Oracle EBS and map it to Siebel CRM interface tables. This process also enables item cross-referencing between Oracle EBS and Siebel CRM.

For more information about initial data loads, see Chapter 2, "Loading Initial Data."

4.1.1 Prerequisites

Find prerequisites in Section 13.3, "Product Management: Prerequisites and Data Requirements."

4.1.2 Solution Assumptions and Constraints

The integration design assumes that the following statements are true:

1. After items and BoMs are synchronized to Siebel CRM, any modifications made to the products or BoMs in Siebel CRM are not synchronized back to Oracle EBS.
2. The initial load must use Oracle Data Integrator (ODI).
3. The following assumptions pertain to the synchronization of BoM structures:
   - All of the BoM product (parent) and BoM components (children) are available in Siebel CRM before creation of the relationship by the BoM synchronization process.
   - Nonorderable items that are part of the product BoM structure are synchronized as part of the BoM synchronization process.
   - Although the relationships between the parent and children are synchronized, the characteristics of the child components, such as Optional, Mandatory and SubAssembly, are not synchronized or maintained in Siebel CRM.
4. There are no mechanism is available to identify which products have a sales BoM structure in Oracle EBS.

   The Siebel product administrator must be made aware of which BoMs that exist in Oracle EBS can be requested for synchronization to Siebel CRM.
5. Table 4–1 illustrates the association that exists between the BOM ITEM TYPE item attribute in Oracle EBS and the STRUCTURE TYPE product attribute in Siebel CRM for synchronizing BoM structures into Siebel and making them configurable and customizable.

Table 4–1 Oracle EBS and Siebel CRM Attribute Association

Oracle EBS BOM Item Type	Siebel CRM Structure Type	BOM Sync Button Enabled
Standard	Bundled	No
Model	Customizable	Yes
Option Class	Customizable	Yes

6. Siebel CRM cannot make requests to synchronize BoM structures for any items in Oracle EBS that have been set up with a BOM ITEM TYPE of *Standard*.

   Because these bundle structures have a static configuration, they are not likely to have a functional usage for Siebel CRM product administrators.
7. Only selected item attributes are synchronized to Siebel CRM. These attributes are supported:
   - Item Name (part of the Key Flex Field structure)
   - Item Description
   - Item Type
   - Item Status

- Primary UOM
- Order Management-related attributes (orderable flag, customer-ordered flag)
- Oracle inventory item ID (required for Oracle Product Configurator)

8. Configured items (* items) are synchronized as part of the product synchronization, but they may include configured items that cannot be ordered from Siebel. For example, an Assemble to Order (ATO) model that is a child component of a Pick to Order ((PTO) model.
9. BoM synchronization must be invoked manually from Siebel CRM, and is not driven by an event in Oracle EBS when a structure is modified or created.
10. Item synchronization is initiated from Oracle EBS as an asynchronous process, and does not include any related item structures.

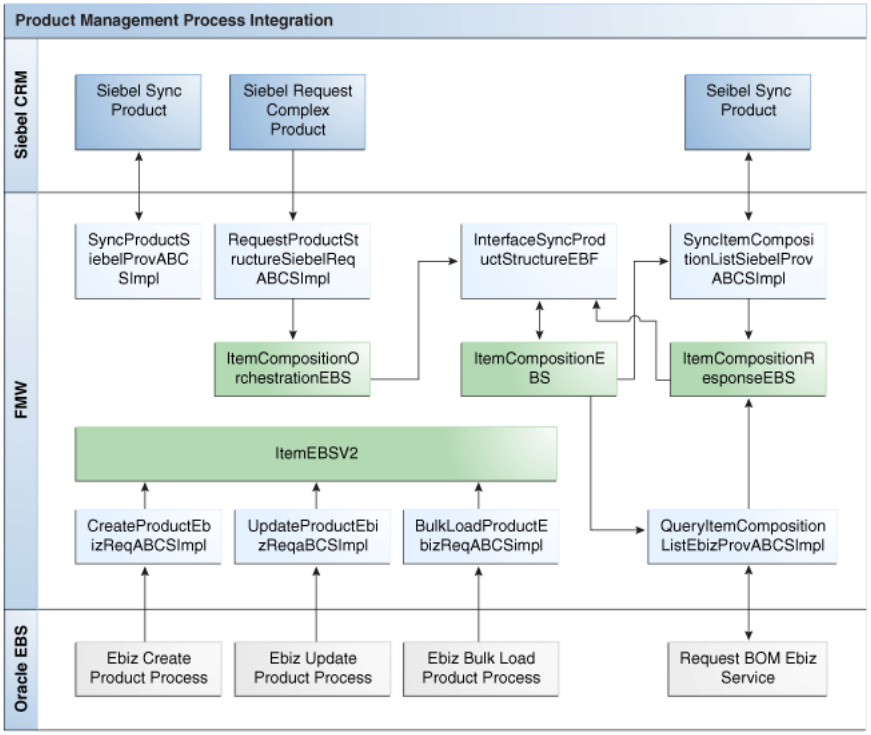
    If the simple product imported into Siebel CRM is associated to BoM structures that must be imported into Siebel CRM, then the product administrator in Siebel does a manual request for BoM from the Product Admin screens.
11. Nonorderable items are not loaded into Siebel CRM as simple products. However, nonorderable items that are part of a BoM structure are loaded into Siebel CRM as component products when a BoM is requested and synchronized to Siebel CRM.
12. If an existing item name in Oracle EBS is updated and synchronized to Siebel CRM, a new product is created in Siebel CRM.
13. A known and identified interdependency exists between the Price Type field values and the Unit of Measure (UOM) field values assigned to an item in Siebel CRM.

    Because price type is not an attribute mapped in the integration between Oracle EBS and Siebel CRM, you must set up a onetime price type in Siebel CRM. You must also explicitly assign any UOM value used in the definition of the item to this price type to ensure that UOM values are synchronized correctly between Oracle EBS and Siebel CRM.
14. Deleting items in Oracle EBS does not result in a request to delete simple products in Siebel CRM because no event mechanisms exist to accomplish this operation.

    Any deletions of simple products in Siebel CRM must be run as a manual process by an administrator in Siebel CRM.
15. Only items associated with Oracle EBS OE Item Validation Org are synchronized from Oracle EBS to Siebel CRM.
16. Oracle does not support custom and user-defined attributes for items or BoMs in this release.

Figure 4–1 illustrates the overall flow for the process integration.

Figure 4–1 Overall Product Management Integration Flow



Oracle EBS is the source for core item information, attributes, and BoM structures. This information is propagated to the Siebel CRM product master to facilitate the order capture process and to enable you to view saleable products in the Siebel CRM application.

Oracle EBS stores the item definitions in these categories:

- Items are childless, single-level structures in Oracle EBS that correspond to simple products in Siebel CRM.

  After a simple product is synchronized in Siebel CRM, any changes to the product attributes or definition in Siebel CRM are not synchronized back to Oracle EBS.

- BoMs are multi-level structures that contain child components and correspond to a complex product in Siebel CRM.

  Complex (configurable) products have a BoM with mandatory components, optional components, or both. After a complex product is synchronized in Siebel CRM, the hierarchical structure of a configurable product is available in Siebel CRM, but updates to the product structure or simple product component attributes are not synchronized back to Oracle EBS.

After products are synchronized in Siebel CRM, product attributes and BoM structures cannot be changed. However, product administrators can leverage eligibility and compatibility rules in Siebel CRM. They can also enrich products with Siebel-specific attributes.

The initial load process ensures that the full array of currently orderable items is synchronized to the Siebel front office application. The set of items synchronized to Siebel CRM include all of the orderable items that can be ordered as standalone items or that function as component (child) items in a complex BoM structure. This initial load process synchronizes only the individual items in a BoM. The relationships or structures of the BoM must be synchronized as an independent process.

Subsequent creation or updates to simple products in the Oracle product master triggers a business event in Oracle EBS that synchronizes these products to Siebel CRM in real time.

Creating or updating BoM structures in Oracle EBS does not automatically trigger a synchronization process to Siebel CRM. Instead, Siebel identifies the complex products to synchronize and invokes a process that allows these structures to be updated in Siebel CRM, based on the most current version of the BoM in the Oracle item master. Siebel CRM users can request a BoM synchronization for multiple BoMs in the same event process.

The complex products synchronized to Siebel CRM cannot be updated, but advanced pricing, eligibility, and compatibility rules can be applied. Any updates to Siebel-specific attributes are not synchronized back to Oracle EBS.

This flow is used for incremental loads; high-volume initial loads should use the Initial Data Load integration flow that is supported by the Order to Cash integration.

For more information about initial data loads, see Chapter 2, "Loading Initial Data."

4.2 Item Synchronization

The Synchronize Items integration flow enables the real-time one-way synchronization of items from Oracle EBS to Siebel CRM. After items are either created or updated in Oracle EBS, a business event is triggered that enables the synchronization of items from Oracle EBS to Siebel CRM.

Items from Oracle EBS are synchronized as products in Siebel CRM:

- For every Item Id and Inventory Org. combination in Oracle EBS where the Inventory Organization also happens to be an Item Validation Organization. Therefore, not all combinations of Item Id and Inventory Org Id from Oracle EBS create a unique Product in Siebel CRM.
- The Siebel CRM product's Business Unit corresponds to the Oracle EBS Operating Unit that has the Inventory Organization as the Item Validation Organization.

  If in Oracle EBS the Item is associated to multiple Operating Units by distinct (non-shared) Item Validation Orgs, then the Item records associated with the Item Validation Orgs in all these multiple Operating Units are synchronized, and cross-references are created. Therefore, in Siebel CRM, the same product is created for different Business Units individually
- Since the Item Validation Org can be shared across multiple operating units in Oracle EBS, all the corresponding business units may have to be associated to the product in Siebel CRM, depending on how product visibility is set up in Siebel CRM. If catalog-based (Enterprise-level) visibility is set up in Siebel CRM, then associating additional business units may not be required. However, if organization-based visibility is set up in Siebel CRM for products, then the additional non-primary (Multi-Org) business unit must be manually associated to the product.
- If the Master Organization is also the Item Validation Org, then these item records with the Master Org Ids are synchronized and cross-references are created. If the Master Org is not the Item Validation Org, then the item records with the Master Org are ignored for creating a cross-reference.

Based on the above, there can be various combinations of Organization set up, and Item and Product definitions across the two applications.

For more information about best practice scenarios for setting up the Organization cross-reference mappings, and how product synchronization to Siebel CRM takes place based on this setup, see Appendix B, "Organization Data Setup for Product Synchronization."

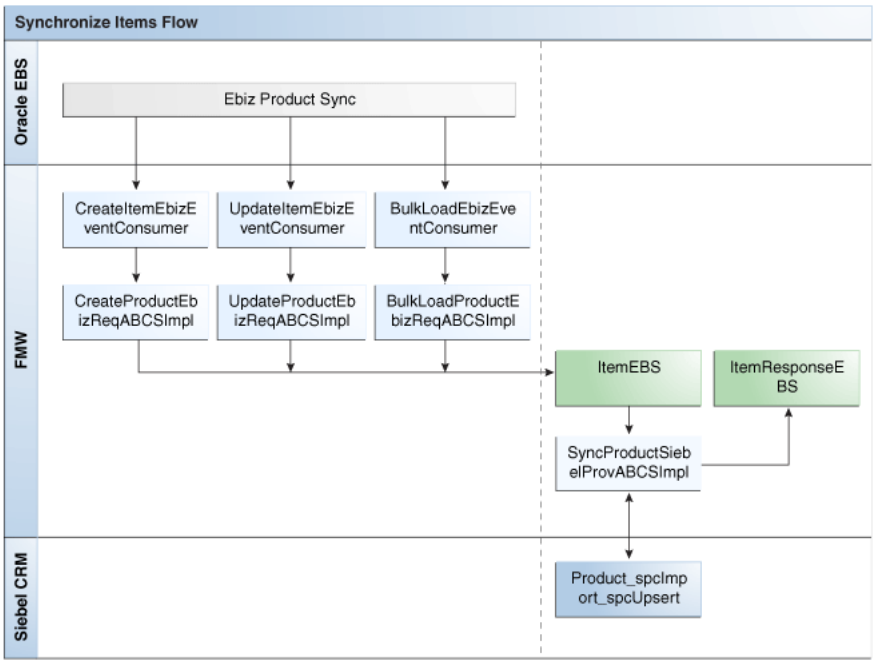
> **Note:** There may be other Item Organization (in Oracle EBS) or Business Unit (in Siebel CRM) related set up options that the implementations make that may not be supported, and may require customization.

Only customer orderable items are synchronized as part of this flow. The following attributes of an item are synchronized and passed to Siebel CRM:

- Item Name (part of the Key Flex Field structure)
- Item Description
- Item Type
- Item Status
- Primary UOM
- Order Management related attributes

Figure 4–2 illustrates the flow for synchronizing items between Oracle EBS and Siebel CRM.

Figure 4–2 Synchronize Items Integration Flow



4.2.1 Create Items Flow

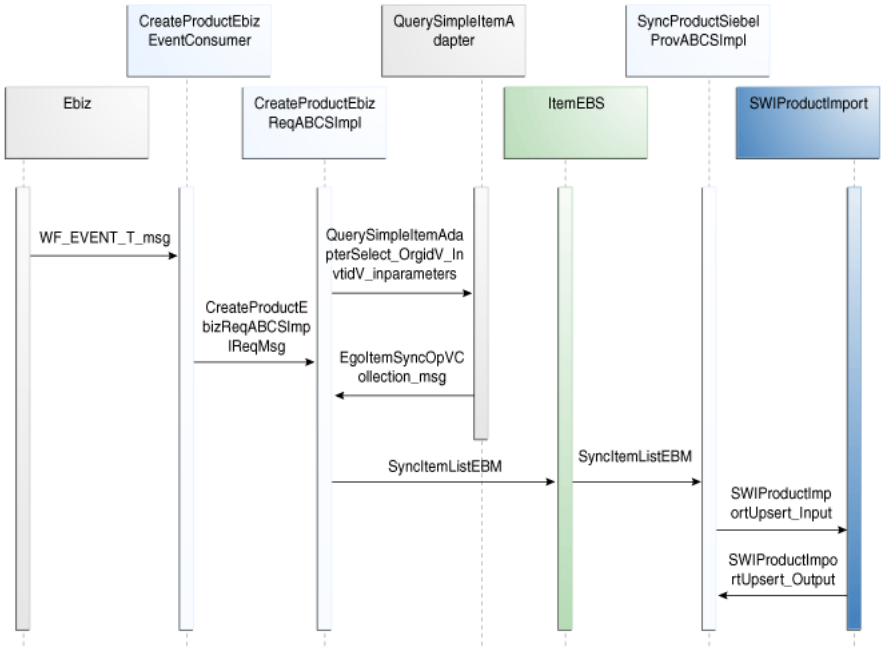
This integration flow uses the following interfaces:

- CreateProductEbizReqABCSImpl

- ItemEBS
- SyncProductSiebelProvABCSImpl
- ItemResponseEBS

Figure 4–3 illustrates the Create Item integration scenario.

Figure 4–3 Create Item Flow Sequence Diagram



When you initiate the process, the following events occur:

1. The CreateItemEbizEventConsumer listens to Business events and receives the WF_EVENT_T_msg event payload for the Create event.

   The CreateItemEbizEventConsumer routes to the CreateProductEbizReqABCSImpl with the complete event payload.

2. The CreateProductEbizReqABCSImpl service calls the Oracle EBS service based on the event payload and then transforms to the SyncItemListEBM and invokes the ItemEBS with the operation SyncItemList.

   - As part of the enrichment process, the CreateProductEbizReqABCSImpl queries the EGO_ITEM_SYNC_OP_V view object through the Oracle EBS adapter QuerySimpleItemAdapter, based on event payload, to get the EgoItemSyncOpVCollection_msg.
   - As part of the transformation process, the CreateProductEbizReqABCSImpl transforms the EgoItemSyncOpVCollection_msg message into the SyncItemListEBM.

     SyncItemListEBM invokes the ItemEBS with the operation SyncItemList. The ItemEBS is a routing service with several operations on the Item enterprise business message (EBM). In the process, the cross-reference is created by concatenation of the Operating Unit and the Inventory Item ID.

3. The ItemEBS routes the SyncItemListEBM to the SyncProductSiebelProvABCSImpl.

4. The SyncProductSiebelProvABCSImpl transforms the SyncItemListEBM to the Siebel Product message and then calls the Siebel product web service SWIProductImport to synchronize the product.

   The Siebel web service completes the request and returns a response message. In the process, the cross-reference is linked to the Siebel IDs of the product.

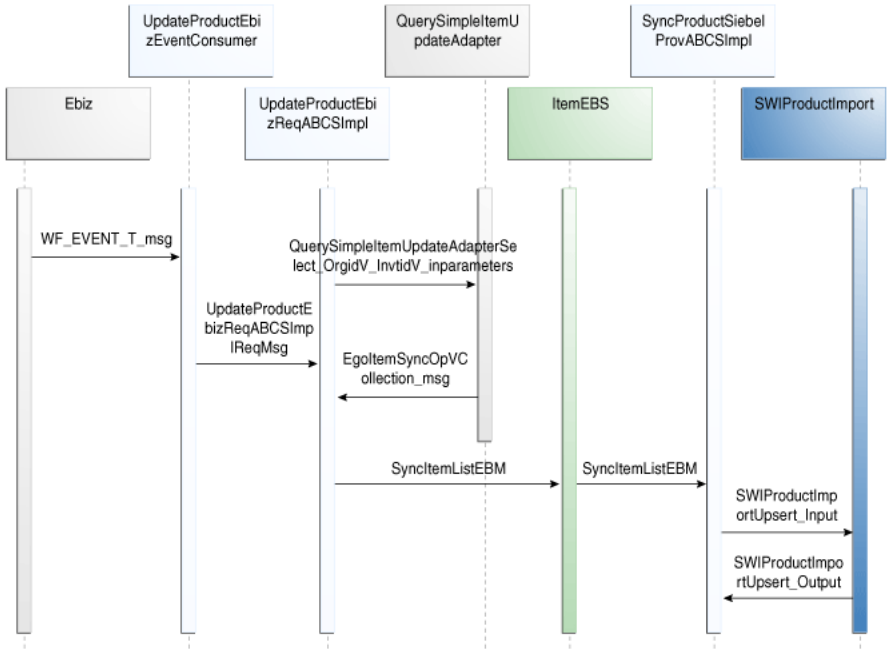
4.2.2 Update Items Integration Flow

This integration flow uses the following interfaces:

- UpdateProductEbizReqABCSImpl
- ItemEBS
- SyncProductSiebelProvABCSImpl

Figure 4–4 illustrates the Update Items integration scenario.

Figure 4–4 Update Items Flow Sequence Diagram



When you initiate the process, the following events occur:

1. The UpdateItemEbizEventConsumer listens to Business events and receives the WF_EVENT_T_msg event payload for the Update event.

   The UpdateItemEbizEventConsumer routes to UpdateProductEbizReqABCSImpl with the complete Event payload.

2. UpdateProductEbizReqABCSImpl service calls the Oracle EBS service based on the event payload, and then transforms to the SyncItemListEBM and invokes the ItemEBS with operation SyncItemList.

   - As part of the enrichment process, the UpdateProductEbizReqABCSImpl queries the EGO_ITEM_SYNC_OP_V view object through the Oracle EBS adapter QuerySimpleItemUpdateAdapter, based on the event payload, to get the EgoItemSyncOpVCollection_msg.

- As part of the transformation process, the UpdateProductEbizReqABCSImpl transforms the EgoItemSyncOpVCollection_msg message into the SyncItemListEBM.
- SyncItemListEBM invokes the ItemEBS with the operation SyncItemList.
- The ItemEBS is a routing service with several operations on the Item EBM.
- In the process, the cross-reference is created by concatenation of the Operating Unit and the Inventory Item ID.

3. The ItemEBS routes the SyncItemListEBM to the SyncProductSiebelProvABCSImpl.
4. The SyncProductSiebelProvABCSImpl transforms the SyncItemListEBM to the Siebel Product message and then calls the Siebel product web service SWIProductImport to synchronize the product.

   The Siebel web service completes the request and returns a response message. In the process, the cross-reference is linked to the Siebel IDs of the product.

4.3 Bill of Material Synchronization

The Synchronize BoM integration flow enables the synchronization of BoMs from Oracle EBS to Siebel CRM. This synchronization is a one-way feed from Oracle EBS to Siebel CRM. Siebel CRM must make a manual request for a structure that provides the applicable product ID for the complex product that must be synchronized. This flow is designed for real time synchronizations; it is not intended for high-volume initial loads.

The Synchronize BoM integration flow brings over all nonorderable items and item relationships based on the requested BoM structure.

The ability to synchronize the structure relies on the existence of the parent item and all of the orderable items in the structure in Siebel CRM at the time the request is made.

This integration flow uses the following interfaces:

- RequestProductStructureSiebelReqABCSImpl
- ItemCompositionOrchestrationEBS
- ItemCompositionEBS
- ItemCompositionResponseEBS
- InterfaceSyncProductStructurelEBF
- QueryItemCompositionListEbizProviderABCSImpl
- SyncItemCompositionListSiebelProvABCSImpl

Figure 4–5 illustrates the flow for synchronizing BoMs from Oracle EBS to Siebel CRM.

Figure 4–5 Synchronize BoM Integration Flow

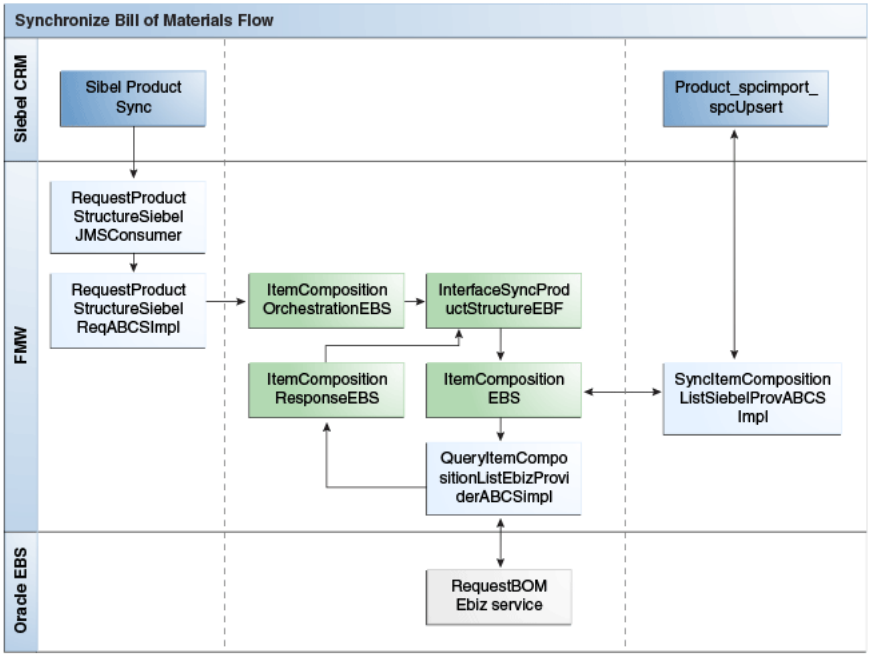
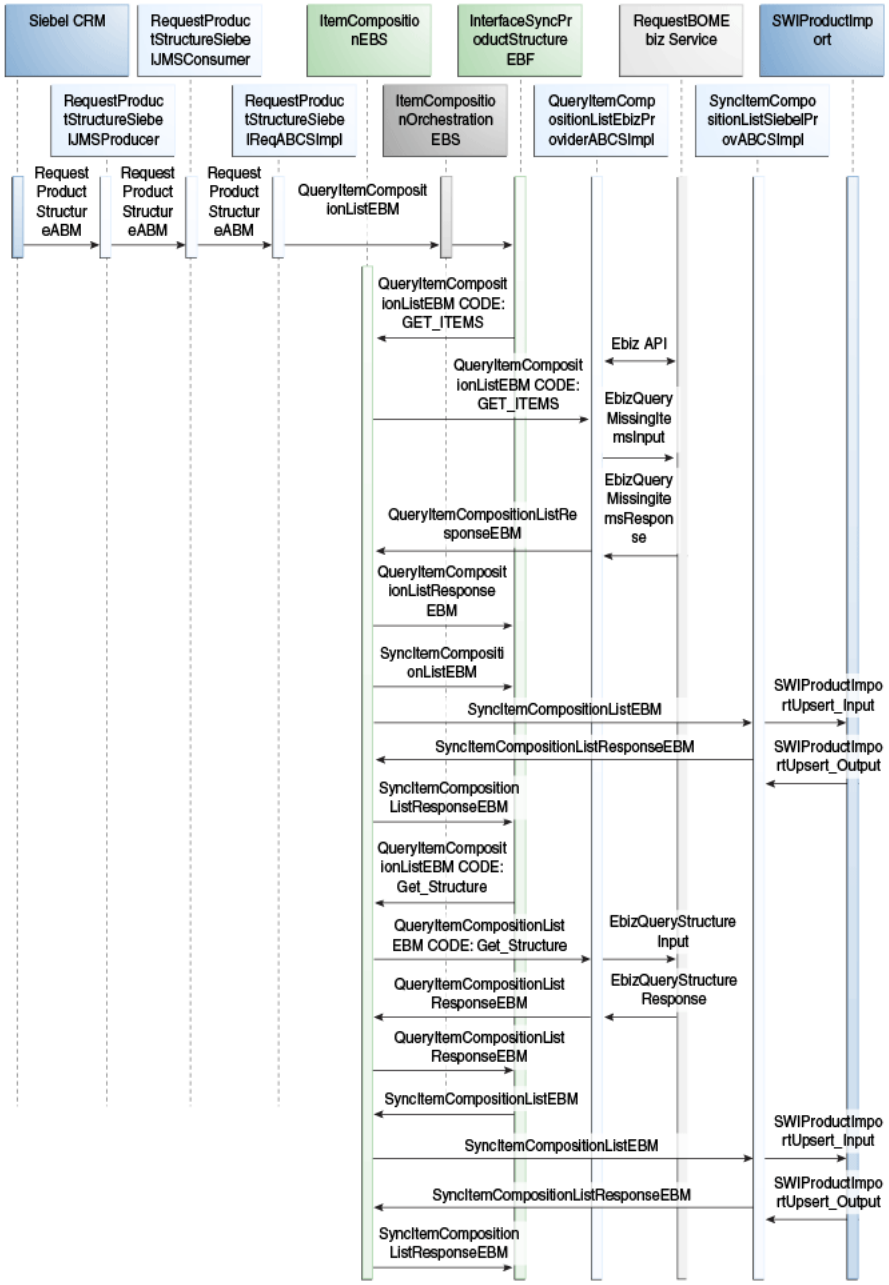


Figure 4–6 illustrates the Synchronize BoM integration scenario.

Figure 4–6 Synchronize BoM Flow Sequence Diagram



When you initiate the process, the following events occur:

1. The SyncProductStructureSiebelJMSProvider enqueues the Product ID, Inventory Location ID, and Business Unit ID to the Java Message Service (JMS) Queue for which the Complex Product and the structure must be available in Siebel CRM.
2. The SyncProductStructureSiebelJMSConsumer process subscribes to the JMS queue and dequeues the Product ID and invokes the RequestProductStructureSiebelReqABCSImpl Siebel requester ABCSImpl.
3. The RequestProductStructureSiebelReqABCSImpl transforms the Siebel product message RequestProductStructureABM into the EBM message

QueryItemCompositionListEBM and invokes the ItemCompositionOrchestrationEBS.

The ItemCompositionOrchestrationEBS is a routing service that routes the request to the InterfaceSyncProductStructurelEBF enterprise business flow.

4. The InterfaceSyncProductStructurelEBF invokes the ItemCompositionEBS with the operation GetComplexProduct

   The ItemCompositionEBS routes the QueryItemCompositionListEBM to the QueryItemCompositionListEbizProviderABCSImpl implementation with EBM Code: GET_ITEMS.

5. The QueryItemCompositionListEbizProviderABCSImpl transforms the QueryItemCompositionListEBM into the Oracle EBS product message EbizQueryMissingItemsInputABM. It then calls the Oracle application processing interface (API) BOM_STRUCT_SYNC_PUB.EXPLODE_STRUCTURE, through the Oracle Apps Adapter, to explode the BOM and query the Oracle View object EGO_ITEM_SYNC_V, BOM_STRUCTURE_SYNC_V through Oracle Apps adapter to fetch nonorderable items from the back office.

   Next, it transforms the Oracle response message EbizQueryMissingItemsResponse into the EBM message QueryItemCompositionListResponseEBM. The QueryItemCompositionListEbizProviderABCSImpl returns a response message QueryItemCompositionListResponseEBM to the ItemCompositionEBS. The ItemCompositionEBS returns a response message, QueryItemCompositionListResponseEBM, to the InterfaceSyncProductStructurelEBF.

6. The InterfaceSyncProductStructurelEBF transforms the QueryItemCompositionListResponseEBM message into the SyncItemCompositionListEBM message and invokes the ItemCompositionEBS with the operation SyncComplexProduct.

   The ItemCompositionEBS routes the SyncItemCompositionListEBM to the SyncProductSiebelProvABCSImpl implementation. In the process, the cross-reference is created by concatenation of the Operating Unit ID and the Inventory Item ID.

7. The SyncProductSiebelProvABCSImpl transforms the SyncItemEBM to the Siebel Product message and then calls the Siebel product web service SWIProductImport to synchronize the product.

   The Siebel web service completes the request and returns a response message. The SyncProductSiebelProvABCSImpl then transforms the Siebel response message to the SyncItemCompositionListResponseEBM and sends it back to the ItemCompositionEBS.

   The ItemCompositionEBS returns the response message SyncItemCompositionListResponseEBM to the InterfaceSyncProductStructurelEBF. In the process, the cross-reference links to the Siebel IDs of the product.

8. The InterfaceSyncProductStructurelEBF invokes the ItemCompositionEBS with the operation GetComplexProduct.

   The ItemCompositionEBS routes the QueryItemCompositionListEBM to the RequestBOMStructureEbizProvABCSImpl implementation with the EBM Code: GET_STRUCTURE.

9. The QueryItemCompositionListEbizProviderABCSImpl transforms the QueryItemCompositionListEBM into the Oracle EBS product message EbizQueryStructureInputABM.

   It then calls the Oracle EBS API BOM_STRUCT_SYNC_PUB.GET_STRUCTURE_PAYLOAD through the Oracle Apps adapter to fetch the Product Structure and transform the Oracle service response message EbizQueryStructureResponseABM into the EBM message QueryItemCompositionListResponseEBM.

   The QueryItemCompositionListEbizProviderABCSImpl returns a response message, QueryItemCompositionListResponseEBM, to the ItemCompositionEBS.

   The ItemCompositionEBS returns a response message, QueryItemCompositionListResponseEBM, to the InterfaceSyncProductStructurelEBF.

10. The InterfaceSyncProductStructurelEBF transforms the QueryItemCompositionListResponseEBM message into the SyncItemCompositionListEBM message and invokes the ItemCompositionEBS with the operation SyncComplexProduct.

    The ItemCompositionEBS routes the SyncItemCompositionListEBM message to the SyncItemCompositionListSiebelProvABCSImpl implementation.

11. The SyncItemCompositionListSiebelProvABCSImpl calls the Siebel web service SWIProductImport to synchronize the item structure.

    The SyncItemCompositionListSiebelProvABCSImpl transforms the SyncItemCompositionListEBM to the Siebel product message and then calls the Siebel product web service SWIProductImport to synchronize the product structure.

    The Siebel web service completes the request and returns a response message. The SyncItemCompositionListSiebelProvABCSImpl then transforms the Siebel response message to the SyncItemCompositionListResponseEBM and sends it back to the ItemCompositionResponseEBS. The ItemCompositionResponseEBS returns a response message, SyncItemCompositionListResponseEBM, to the InterfaceSyncProductStructurelEBF.

4.4 Siebel CRM Interfaces

For the Product Management integration flow, these are the Siebel CRM web services:

Inbound Siebel CRM Web Service

- Logical name: SiebelProductService.upsert
- Service Name: SWIProductImport (maps to ProductIntegration)
- Operation Name: SWIProductImportUpsert (maps to Upsert)
- Request Schema: SWIProductIntegrationIO.xsd
- Response Schema: SWIProductIntegrationIO.xsd

Outbound Siebel CRM Web Service

- Event: Request BOM
- Request Schema: SWIProductIntegrationIOReq.xsd

- Expected Event Action: Siebel calls the RequestProductStructureSiebelJMSProducer service with an application business message (ABM). The ABM definition includes a list of product IDs.

For more information about Siebel web services, see *CRM Web Services Reference*.

4.5 Oracle EBS Interfaces

For the Product Management integration flow, these are the Oracle EBS web services:

Inbound to Oracle EBS Web Services

- Service Name: BOM_STRUCT_SYNC_PUB.EXPLODE_STRUCTURE
- Service Name: Using database adapter generates a partner link joining the following two views: EGO_ITEM_SYNC_V and BOM_STRUCTURE_SYNC_V.

  The filter conditions are:

  - BOM_STRUCTURE_SYNC_V.COMP_CUSORDER_FLAG = 'N'
  - BOM_STRUCTURE_SYNC_V.TOP_ITEM_ID=&top_item_id
  - BOM_STRUCTURE_SYNC_V.ORGANIZATION_ID= & ORGANIZATION_ID
- Service Name: BOM_STRUCT_SYNC_PUB.GET_STRUCTURE_PAYLOAD
- Service Name: EGO_ITEM_SYNC_V

Outbound from Oracle EBS Event Interfaces

- oracle.apps.ego.item.postItemCreate
  - Event Name: Inventory Item ID
  - Event Name: Inventory Organization ID
- oracle.apps.ego.item.postItemUpdate
  - Event Name: Inventory Item ID
  - Event Name: Inventory Organization ID
- oracle.apps.ego.item.postItemBulkload
  - Event Name: Request ID

For more information about Oracle E-Business Suite web services and documentation prior to Release 12.1.3, see the library on Oracle Technology Network: http://www.oracle.com/technetwork/documentation/applications-167706.html?. For Oracle E-Business Suite documentation for R12.1.3 and beyond, see this library: http://download.oracle.com/docs/cd/E18727_01/index.htm?

4.6 Core Oracle AIA Components

The Product Management integration flow uses the following delivered core components:

- ItemCompositionEBO
- ItemCompositionListEBM
- ItemEBO
- ItemListEBM

- ItemEBSV2
- ItemCompositionOrchestrationEBS
- InterfaceSyncProductStructureEBF
- ItemCompositionEBS
- ItemCompositionResponseEBS

The industry enterprise business object (EBO) and enterprise business message XML schema (EBM XSD) files are located here: $AIA_HOME/AIAMetaData/AIAComponents/EnterpriseObjectLibrary/Core/EBO/

The industry enterprise business service (EBS) WSDL files are located here: $AIA_HOME/AIAMetaData/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/

For detailed documentation of individual EBOs and EBMs, click the AIA Reference Doc link on EBO and EBM detail pages in Oracle Enterprise Repository.

For more information about using the Oracle Enterprise Repository and configuring it to provide the AIA Reference Doc link, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring and Using Oracle Enterprise Repository as the Oracle AIA SOA Repository."

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

For more information, see *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Extensibility."

4.7 Integration Services

These services are delivered with this integration:

- ItemEBSV2
- ItemCompositionResponseEBSV2
- ItemResponseEBSV2
- CreateProductEbizReqABCSImpl
- UpdateProductEbizReqABCSImpl
- BulkLoadProductEbizReqABCSImpl
- SyncProductSiebelProvABCSImpl
- RequestProductStructureSiebelReqABCSImpl
- ItemCompositionOrchestrationEBS
- InterfaceSyncProductStructureEBF
- QueryItemCompositionListEbizProvABCSImpl
- ItemCompositionEBS
- SyncItemCompositionListSiebelProvABCSImpl
- RequestProductStructureSiebelJMSProducer
- RequestProductStructureSiebelJMSConsumer
- CreateItemEbizEventConsumer

- UpdateItemEbizEventConsumer
- BulkloadItemEbizEventConsumer

For more information, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring and Using Oracle Enterprise Repository as the Oracle AIA SOA Repository."

4.7.1 ItemEBSV2

The ItemEBSV2 is a lightweight routing service that exposes all of the enterprise operations that can be performed on an ItemEBO.

The SyncItemList operation is provided by the ItemEBS.

SyncItem (Simple Product): Whenever an Item is created or updated in the Item master in Oracle EBS; a business event is raised to sync the item in Siebel CRM.

The synchronization is required only for items that belong to OE Item Validation Org, are customer orderable, and have Item Type = {Model, Option Class, and Standard}. The business event does not discriminate between items based on this mentioned constraint-the event is triggered regardless of these constraints.

Post Item Create: oracle.apps.ego.item.postItemCreate

Post Item Update: oracle.apps.ego.item.postItemUpdate

Post Item BulkLoad: Oracle.apps.ego.item.postItemBulkload

For more information about this EBS, see the *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Designing and Developing Enterprise Business Services" and *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Enterprise Business Services."

4.7.2 ItemCompositionResponseEBSV2

The ItemCompositionResponseEBSV2 is a lightweight routing service that exposes all of the enterprise response operations that can be performed with an ItemComposition enterprise object. All of the Order to Cash complex product integration flows make use of the response operations provided by this enterprise business service. This service is provided as a front end to other implementation services.

ItemCompositionResponseEBSV2 provides the SyncItemCompositionListResponse operation.

4.7.3 ItemResponseEBSV2

The ItemResponseEBSV2 is a lightweight routing service that exposes all of the enterprise response operations that can be performed with an Item enterprise object. As delivered, no routing rules are defined in the ItemResponseEBS. It is provided for future use.

4.7.4 CreateProductEbizReqABCSImpl

The CreateProductEbizReqABCSImpl is a Business Process Execution Language (BPEL) process that is responsible for calling the Oracle EBS Product web service, based on the event payload, to get the Oracle EBS Product ABM, and for transforming the Oracle EBS Product ABM messages into the appropriate Item EBM format and invoking the SyncItem EBS operation. By default, an orderable product associated

with the OE:Item Validation Org is synchronized to Siebel CRM. To bypass validation against the OE:Item Validation Org, the configuration property BYPASS_ITEMVALIDATIONORG_FLAG must be set to Y.

This is a single operation service. It accepts an Oracle EBS Product message as a request and does not return a response.

4.7.5 UpdateProductEbizReqABCSImpl

The UpdateProductEbizReqABCSImpl is a BPEL process that is responsible for calling the Oracle EBS Product web service, based on the event payload to get the Oracle EBS Product ABM and transforming the Oracle EBS Product ABM messages into the appropriate Item EBM format and invoking the SyncItemList EBS operation. By default, an Orderable Product associated with the OE:Item validation organization is synchronized to Siebel CRM. To bypass validation against the OE:Item Validation Org, the configuration property BYPASS_ITEMVALIDATIONORG_FLAG must be set to Y.

This is a single operation service. It accepts an Oracle EBS Product message as a request and does not return a response.

4.7.6 BulkLoadProductEbizReqABCSImpl

The BulkLoadProductEbizReqABCSImpl is a BPEL process that is responsible for calling the Oracle EBS Product web service, based on the event payload, to get the Oracle EBS Product ABM and transforming the Oracle EBS Product ABM messages into the appropriate Item EBM format and invoking the SyncItemList EBS operation. By default, an orderable product associated with the OE:Item Validation Org is synchronized to Siebel CRM. To bypass validation against OE:Item Validation Org, the configuration property BYPASS_ITEMVALIDATIONORG_FLAG must be set to Y.

This is a single operation service. It accepts an Oracle EBS Product message as a request and does not return a response.

4.7.7 SyncProductSiebelProvABCSImpl

The SyncProductSiebelProvABCSImpl is a BPEL process that receives the ItemEBM and transforms it into a Siebel Product ABM. It invokes the Siebel Product web service to synchronize the same into Siebel. The Model and Option class item is synchronized to Siebel CRM as a customizable product that does not get released, whereas a standard item is synchronized to Siebel CRM as a bundle product and is released as part of the synchronize process.

The configuration property controls the released product in Siebel CRM:

- SIMPLE_PRODUCT_RELEASE_FLAG: Default value Y
- COMPLEX_PRODUCT_RELEASE_FLAG: Default value N

This is a single operation service. It accepts a SyncItemListEBM Product message as a request.

4.7.8 RequestProductStructureSiebelReqABCSImpl

The RequestProductStructureSiebelReqABCSImpl is a BPEL process that receives the Siebel product message RequestProductStructureABM from the SyncProductStructureSiebelJMSConsumer service and is responsible for transforming the RequestProductStructureABM into the QueryItemCompositionListEBM message and invoking the ItemCompositionOrchestrationEBS.

This asynchronous service accepts the Siebel product message RequestProductStructureABM as a request and does not return a response.

4.7.9 ItemCompositionOrchestrationEBS

The ItemCompositionOrchestrationEBS is a lightweight routing service that routes the request from the RequestProductStructureSiebelReqABCSImpl service to the InterfaceSyncProductStructurelEBF Enterprise Business Flow service. The ItemCompositionOrchestrationEBS does not expose any enterprise operations that can be performed with an Item Composition enterprise object.

The operation ProcessItemComposition is provided by the ItemCompositionOrchestrationEBS enterprise business service for routing purposes.

4.7.10 InterfaceSyncProductStructureEBF

The InterfaceSyncProductStructureEBF enterprise business flow service is an asynchronous BPEL process that queries the Non Orderable (default) or All (based on the configuration property PUBLISH_ALL_PRODUCT) products that are associated with the complex product and synchronizes them to Siebel CRM. This service is invoked from the ItemCompositionOrchestrationEBS service with request message QueryItemCompositionListEBM.

A single operation is available for this service, and the input is an instance of the QueryItemCompositionListEBM message. The Data Area of the message contains only one Product ID, Inventory Location ID, and Business Unit ID. (In the case of multiple product requests from Siebel, the service iterates through the collection in the RequestProductStructureSiebelReqABCSImpl and invokes the InterfaceSyncProductStructureEBF for each complex product.)

4.7.11 QueryItemCompositionListEbizProvABCSImpl

The InterfaceSyncProductStructureEBF invokes the QueryItemCompositionListEbizProvABCSImpl BPEL process through the ItemCompositionEBS. This service is responsible for transforming the QueryItemCompositionListEBM into the EbizQueryMissingItemsInputABM Oracle product message and running Oracle API BOM_STRUCT_SYNC_PUB.EXPLODE_ STRUCTURE through Oracle Apps adapter to explode the BOM.

By default, the QueryItemCompositionListEbizProvABCSImpl publishes all nonorderable child components of the BOM and BOM Structure. The configuration property PUBLISH_ALL_PRODUCT controls this function.

This is a single operation service that accepts a QueryItemCompositionListEBM message as a request and returns a QueryItemCompositionListResponseEBM as a response.

4.7.12 ItemCompositionEBS

The ItemCompositionEBS is a lightweight routing service that exposes all of the enterprise operations that can be performed with an ItemComposition EBO.

ItemCompositionEBS provides these operations:

- QueryItemCompositionList
- SyncItemCompositionList

4.7.12.1 QueryItemCompositionList

The ItemCompositionEBS calls the QueryItemCompositionListEbizProviderABCSImpl Oracle implementation to get item and item structure. The QueryItemCompositionListEbizProviderABCSImpl runs the following Oracle APIs:

- BOM_STRUCT_SYNC_PUB.EXPLODE_STRUCTURE to explode the BOM
- Query Oracle View object EGO_ITEM_SYNC_V, BOM_STRUCTURE_SYNC_V to fetch the Non Orderable Item (default) or All (based on the configuration property PUBLISH_ALL_PRODUCT)
- BOM_STRUCT_SYNC_PUB.GET_STRUCTURE_PAYLOAD to get the BOM structure

4.7.12.2 SyncItemCompositionList

When a Siebel product administrator requests one or more product structures to be available in the Siebel CRM system, the product and its corresponding product structure are synchronized through the ItemCompositionEBS from Oracle EBS back into Siebel CRM.

The ItemCompositionEBS calls the SyncItemCompositionListSiebelProvABCSImpl Siebel implementation to synchronize the product and corresponding product structure into Siebel CRM.

4.7.13 SyncItemCompositionListSiebelProvABCSImpl

The ItemCompositionEBS invokes the SyncItemCompositionListSiebelProvABCSImpl Siebel implementation. The responsibility of this service is to receive SyncItemCompositionListEBM, transform it into a Siebel Product ABM, and then invoke the Siebel Product web service to synchronize the item structure into Siebel CRM.

Model and Option class items are synchronized to Siebel as customizable products and do not get released, whereas a standard item is synchronized to Siebel as Bundle Product and released as part of synchronization process.

The following configuration properties control the Release Product in Siebel:

- SIMPLE_PRODUCT_RELEASE_FLAG: Default value Y
- COMPLEX_PRODUCT_RELEASE_FLAG: Default value N

This is a single operation service that accepts a SyncItemCompositionListEBM Product message as a request and invokes the ItemCompositionResponseEBS with the operation SyncItemCompositionListResponse and the message SyncItemCompositionListResponseEBM to return the response message to the InterfaceSyncProductStructureEBF.

4.7.14 RequestProductStructureSiebelJMSProducer

The RequestProductStructureSiebelJMSProducer is implemented as a BPEL process because it involves JMS Header manipulations.

This service is responsible for enqueueing the Siebel RequestProductStructure ABM event payload into the JMS queue when a Siebel product administrator decides to replicate an existing complex product from the back office in Siebel. The enqueued message can be a list of product IDs with the related inventory location IDs and business unit IDs.

This service has one asynchronous request-only operation: RequestProductStructure

4.7.15 RequestProductStructureSiebelJMSConsumer

The RequestProductStructureSiebelJMSConsumer is implemented as a process with JMS adapter and routing services.

This service is responsible for dequeueing the Siebel RequestProductStructureABM message from the JMS queue AIA_ITEMCOMPJMSQUEUE and invoking the RequestProductStructure operation of the RequestProductStructureSiebelReqABCSImpl.

One service with one operation is available for reading Siebel messages RequestProductStructureMsg from the JMS queue AIA_ITEMCOMPJMSQUEUE. That service is RequestProductStructureJMSConsumer

4.7.16 CreateItemEbizEventConsumer

The CreateItemEbizEventConsumer has an Oracle Apps adapter configured to listen for create business events with routing services.

One service with one operation is available to read the Oracle EBS message EbizSyncProductReqMsg from the Oracle AQ WF_BPEL_Q. That service is CreateItemEbizEventConsumer

This service is implemented as a process with an Oracle Apps adapter for listening to business event and routing services.

4.7.17 UpdateItemEbizEventConsumer

The UpdateItemEbizEventConsumer has an Oracle Apps adapter configured to listen for update business events with routing services.

One service with one operation is available to read the Oracle EBS message EbizSyncProductReqMsg from the Oracle AQ WF_BPEL_Q. That service is UpdateItemEbizEventConsumer

This service is implemented as a process with an Oracle Apps adapter for listening to business events and routing services.

4.7.18 BulkloadItemEbizEventConsumer

The BulkloadItemEbizEventConsumer has an Oracle Apps adapter configured to listen for Bulkload business events with routing services.

One service with one operation is available to read the Oracle EBS message EbizSyncProductReqMsg from the Oracle AQ WF_BPEL_Q. That service is BulkloadItemEbizEventConsumer

This service is implemented as a process with an Oracle Apps adapter for listening to business events and routing services.

5

Process Integration for Price Lists

This chapter provides an overview of the process integration for price lists and discusses loading initial and incremental price lists, and solution assumptions and constraints.

This chapter includes the following section:

5.1 Process Integration for Price Lists

The process integration for price lists is different from the other integrations in the Order to Cash pre-built integration. The price list integration provides an initial bulk load process and an incremental load process that both use Oracle Data Integrator (ODI) to update price list data from Oracle E-Business Suite (Oracle EBS) to Siebel Customer Relationship Management (Siebel CRM).

The price list integration between Oracle EBS and Siebel CRM supports the following integration flows:

- **Initial or bulk load:** This flow enables the extract, transformation, and load (ETL) of initial price list data from Oracle EBS to Siebel CRM.

  This feature uses ODI to extract relevant data from Oracle EBS and map it to Siebel CRM interface tables. This process also enables cross-referencing between Oracle EBS and Siebel CRM.

- **Incremental load:** This flow moves new price lists and lines or updates to existing price lists from Oracle EBS into Siebel CRM for use in the order capture process.

For more information about initial data loads, see Chapter 2, "Loading Initial Data."

For more information about incremental loads, see Chapter 14.5, "Loading Initial Price List Data"

The price list integration point is different from that of other data types, such as customer and product. Use the bulk load feature to synchronize changes to active price list data in the Oracle EBS database to the Siebel CRM database. Changes after the initial bulk load can be either the creation of new price lists or updates to the header and line information of existing price lists.

The incremental bulk loads create or update price lists or the header and line information in the Siebel CRM database, depending on whether the price lists or the header and line information exist in that database.

If you change the name of a price list in Oracle EBS, a new price list with that name is created in Siebel CRM. Siebel CRM does not delete or rename the original price list. You can do that manually if required.

If Pricing Security is turned on and if a price list is not global, you must initialize the context in Oracle EBS before synchronizing price list lines.

To initialize, use the QP_CRMINTEG_PRICELIST_V view and the following statement:

```
fnd_global.apps_initialize(<User ID>, <Responsibility ID>,<Application id>);
For example: fnd_global.apps_initialize(1318, 24021, 661);
```

To make a price list nonglobal, clear the Global check box on the Oracle EBS price list screen. You can use a nonglobal price list only in the operating unit set in the Oracle EBS profile `MO: Operating Unit`.

If you change the start date of a price list in Oracle EBS, the system creates a new price list with the same name in the Siebel database. However, if you change only the end date of a price list in Oracle EBS, the system updates the existing price list in the Siebel database.

5.1.1 Prerequisites

Find prerequisites in Section 13.4, "Price Lists: Prerequisites."

5.1.2 Solution Assumptions and Constraints

The integration design assumes that the following statements are true:

1. Only a bulk load process is supported.
   - ODI pulls all of the order-related active price lists from Oracle EBS into Siebel CRM.
   - To synchronize incremental loads, run the same process but specify the price list names that pull those price lists and lines from Oracle EBS into Siebel CRM, including any changes to those price lists. The process updates or creates price lists and lines, depending on whether the price list and lines exist in Siebel CRM.
2. Order-capture related active price lists from Oracle EBS to Siebel CRM are synchronized as part of the initial load.
3. Price lists such as those used by purchasing, interoffice, contracts, and so on are not synchronized.
4. Price list header information (such as name, currency, start date, and end date), and price list line information (such as product, list price, start date, and end date) are synchronized.

   Advanced pricing features, such as volume discount, attribute pricing, qualifier, and so on, are not synchronized.

For more information about initial data loads, see Chapter 2, "Loading Initial Data."

For more information about incremental loads, see Chapter 14.5, "Loading Initial Price List Data"

6

Process Integration for Quotes

This chapter provides an overview of the process integration for Quotes and discusses the Quotes integration flow, Siebel Customer Relationship Management (Siebel CRM) and Oracle E-Business Suite (Oracle EBS) interfaces, core Oracle Application Integration Architecture (Oracle AIA) components, and integration services.

This chapter includes the following sections:

6.1 Process Integration for Quotes

The process integration for Quotes enables companies to negotiate sales with prospects based on a variety of factors, such as product and price. Siebel CRM is used to capture quotes. Because Siebel CRM also handles pricing, quotes can be created and displayed immediately. After a quote is approved, it can be sent to the back office for order fulfillment. After a quote is submitted to the back office, no further updates can be made in Siebel CRM.

The Quote integration flow can enable the Customer integration flow if required. If a quote has customer information that has not yet been created in the back office, the customer information is synchronized before the quote is created in the back office.

Quotes can consist of simple products or configurable products. When a quote has configurable products, they are customized by invoking the Oracle Configurator. Siebel CRM and Oracle Configurator are seamlessly integrated.

For more information about Oracle Configurator, see that product's documentation

6.1.1 Prerequisites

Find prerequisites in Section 13.5, "Quotes: Prerequisites and Data Requirements."

6.1.2 Solution Assumptions and Constraints

The constraints for the Quotes integration are:

1. When a quote is submitted to Oracle EBS, the quote integration life cycle ends. There are no further updates to it in Siebel CRM.
2. Quotes must be in approved status before they can be submitted to the back office.
3. Business-to-Customer (B2C) scenarios are not supported by this integration.
4. As delivered, available-to-promise (ATP) check, credit check, and shipping charges services are not supported with Quotes.

6.2 Quotes Integration Flow

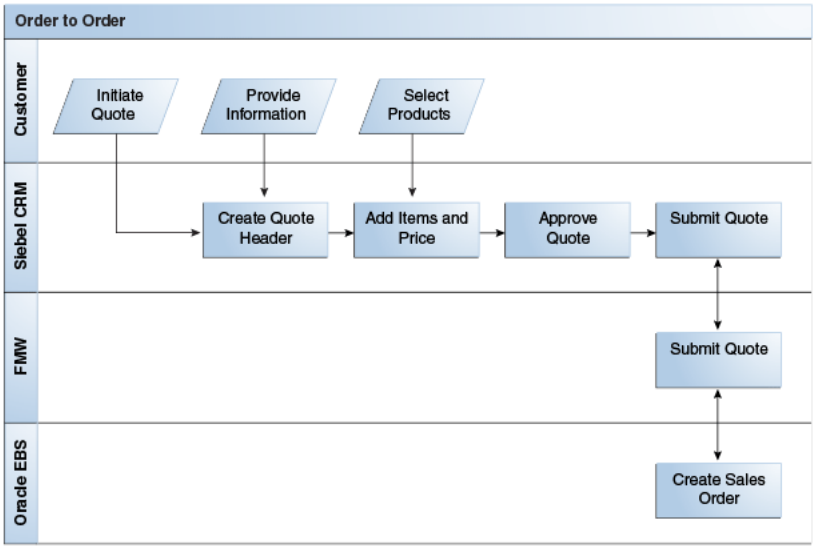
In the Quotes integration flow, a customer initiates a request for a quote. The customer service representative (CSR) captures all of the necessary header information and then proceeds with the quote line items, based on the products requested by the customer.

Siebel CRM is used to determine pricing. Siebel CRM also allows the immediate creation of quotes without getting pricing information from a back office system.

The CSR gives the quote details to the customer. After the customer approves the quote, the CSR submits it to the back office for order fulfillment.

Figure 6–1 illustrates the flow for the Quotes process integration.

Figure 6–1 Quotes Process Integration



6.3 Siebel CRM Interfaces

The Quotes integration flow uses the following Siebel CRM web services:

Inbound Siebel CRM Web Services

- Service Name: SWIQuoteUpsert
- Operation Name: SBLQuoteUpsert
- Request Schema: SWIQuoteIO.xsd
- Response Schema: SWIQuoteIO.xsd

Outbound Siebel CRM Web Service

- Quote Submitted - Siebel invokes the ProcessQuoteSiebelJMSProducer with the ListOfSWIQuoteIO application business message (ABM).

For more information about Siebel web services, see *CRM Web Services Reference*.

6.4 Oracle EBS Interfaces

The Quotes integration flow uses the following Oracle EBS web service:

Inbound to EBS Web Service

- OE_INBOUND_INT.PROCESS_ORDER (Process Sales Order Service)

For more information about Oracle E-Business Suite web services and documentation prior to Release 12.1.3, see the library on Oracle Technology Network: http://www.oracle.com/technetwork/documentation/applications-167706.html?. For Oracle E-Business Suite documentation for R12.1.3 and beyond, see this library: http://download.oracle.com/docs/cd/E18727_01/index.htm?

6.5 Core Oracle AIA Components

The Quotes integration uses the following delivered horizontal components:

- SalesOrderEBO
- ProcessSalesOrderEBM
- ProcessSalesOrderResponseEBM
- CreateSalesOrderEBM
- CreateSalesOrderResponseEBM
- UpdateSalesOrderEBM
- UpdateSalesOrderResponseEBM

The industry enterprise business object (EBO) and enterprise business message XML schema (EBM XSD) files are located here: $AIA_HOME/AIAMetaData/AIAComponents/EnterpriseObjectLibrary/Core/EBO/

The industry enterprise business service (EBS) WSDL files are located here: $AIA_HOME/AIAMetaData/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/

For detailed documentation of individual EBOs and EBMs, click the AIA Reference Doc link on EBO and EBM detail pages in Oracle Enterprise Repository.

For more information about using the Oracle Enterprise Repository and configuring it to provide the AIA Reference Doc link, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring and Using Oracle Enterprise Repository as the Oracle AIA SOA Repository."

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

For more information, see *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Extensibility."

6.6 Integration Services

The following services are delivered with the process integration for Quotes:

- ProcessQuoteSiebelJMSProducer
- ProcessQuoteSiebelJMSConsumer
- ProcessQuoteSiebelReqABCSImpl
- ProcessQuoteSoapMsgSiebelJMSConsumer

Additionally, the process integration for Quotes uses the following services from the process integration for Orders:

- SalesOrderOrchestrationEBSV2
- SalesOrderOrchestrationResponseEBSV2
- InterfaceSalesOrderToFulfillmentEBF
- InterfaceSalesOrderToCustomerEBFV2
- SalesOrderEBSV2
- CreateSalesOrderEbizProvABCSImpl
- ProcessSalesOrderEbizAdapter
- SalesOrderResponseEBSV2
- UpdateSalesOrderSiebelProvABCSImpl

For more information, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring and Using Oracle Enterprise Repository as the Oracle AIA SOA Repository."

6.6.1 ProcessQuoteSiebelJMSProducer

The ProcessQuoteSiebelJMSProducer service is responsible for providing guaranteed delivery of the ProcessQuote Siebel ABM to the Java Message Service (JMS) queue. This service is invoked synchronously from the Siebel application workflow. The response indicates whether the message was successfully enqueued. This service is invoked as part of the Create Quote integration flow only when Siebel application 8.0.0.x versions are used.

You should assume that the Siebel Quote was validated from within the Siebel workflow.

The ProcessSalesQuoteSiebelJMSProducer service has a single synchronous request+reply operation: ProcessQuote.

6.6.2 ProcessQuoteSiebelJMSConsumer

The ProcessQuoteSiebelJMSConsumer service dequeues the ProcessQuote Siebel ABM from the JMS queue and asynchronously invokes the ProcessQuoteSiebelReqABCSImpl service. This service is invoked as part of the Create Quote integration flow only when the Siebel application 8.0.0.x versions are used.

The ProcessQuoteSiebelJMSConsumer service has an inbound JMS adapter front end that subscribes to the JMS queue.

The ProcessQuoteSiebelJMSConsumer service is implemented as an inbound JMS adapter service in Mediator.

6.6.3 ProcessQuoteSoapMsgSiebelJMSConsumer

The ProcessQuoteSoapMsgSiebelJMSConsumer service dequeues the ProcessQuote Siebel ABM from the JMS queue and asynchronously invokes the ProcessQuoteSiebelReqABCSImpl service. This service is invoked as part of the Create Quote integration flow when the version of the Siebel CRM application is 8.1.1.x.

The ProcessQuoteSoapMsgSiebelJMSConsumer service has an inbound JMS adapter front end that subscribes to the JMS queue.

The ProcessQuoteSoapMsgSiebelJMSConsumer service is implemented as an inbound JMS adapter service in Mediator.

6.6.4 ProcessQuoteSiebelReqABCSImpl

The ProcessQuoteSiebelReqABCSImpl service transforms the ProcessQuote Siebel ABM into the canonical ProcessSalesOrderEBM and asynchronously invokes the ProcessSalesOrder operation of the SalesOrderOrchestrationEBSV2 to initiate the InterfaceSalesOrderToFulfillmentEBFV2. This service is invoked as part of the Create Order/Quote integration flow.

As part of the transformation to the ProcessSalesOrderEBM, the system generates common IDs for the quote and quote lines, referenced accounts, referenced ship-to and bill-to addresses, and contacts, and populates the cross-reference with them.

The ProcessQuoteSiebelReqABCSImpl service has a single asynchronous request-only operation, ProcessQuote, and it accepts the Siebel Quote ABM.

The one transformation is ProcessQuoteABM to ProcessSalesOrderEBM.

The ProcessQuoteSiebelReqABCSImpl application business connector service (ABCS) is implemented as an asynchronous request-only Business Process Execution Language (BPEL) process.

7

Available to Promise Check Integration Flow

This chapter provides an overview of the Available to Promise (ATP) Check integration flow and discusses Siebel Customer Relationship Management (Siebel CRM) and Oracle E-Business Suite (Oracle EBS) interfaces, core Oracle Application Integration Architecture (Oracle AIA) components, and integration services.

This chapter includes the following sections:

7.1 ATP Check Requests

The ATP Check integration flow can be called on request before the Create (or Revise) Order integration flow. The ATP Check integration flow starts in Siebel CRM to obtain product availability quantities and dates from Oracle EBS. This data is returned to Siebel CRM so that a Customer Service Representative (CSR) can inform the customer and continue with the order submit process. Calling ATP Check before an order is submitted increases the chances of successfully fulfilling the order from the back office. Setting correct expectations with customers increases customer satisfaction.

The product Id, requested date, and quantities are sent to Oracle EBS to check the available inventory. Oracle EBS can return a promise date and quantity, or it can present multiple promise dates if the desired quantity is not available on the requested date. ATP information can be requested for an order line or for an entire order. Reservations are not supported.

> **Note:** Inventory locations must be maintained manually in both systems and they must have the same name and address. Inventory Location Ids are manually cross-referenced.

This integration flow works only with Siebel sales orders, not quotes.

Figure 7–1 and Figure 7–2 illustrate where the ATP Check integration flow fits in the Order to Cash integration flow.

Figure 7–1 ATP Check Integration Flow (1 of 2)

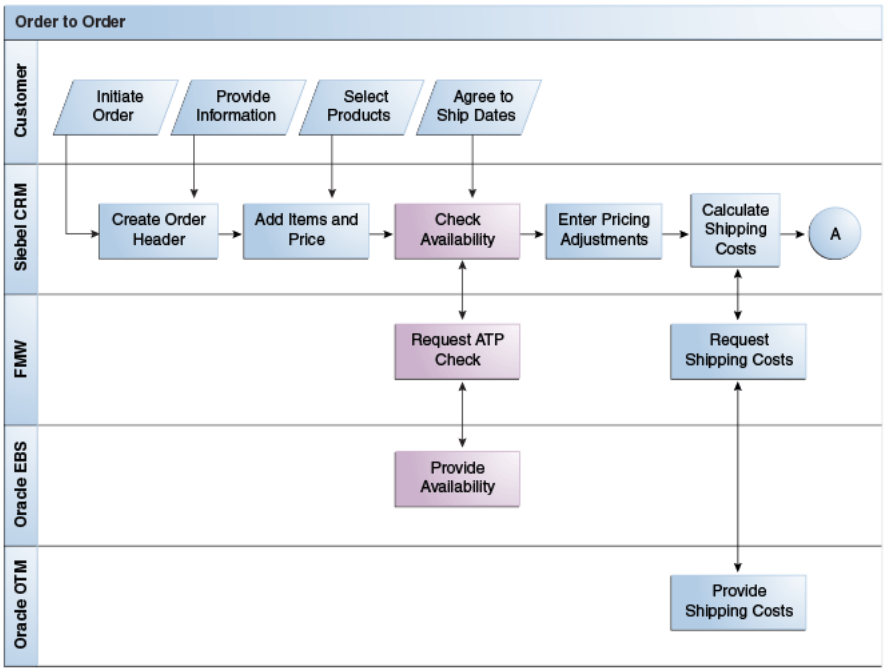
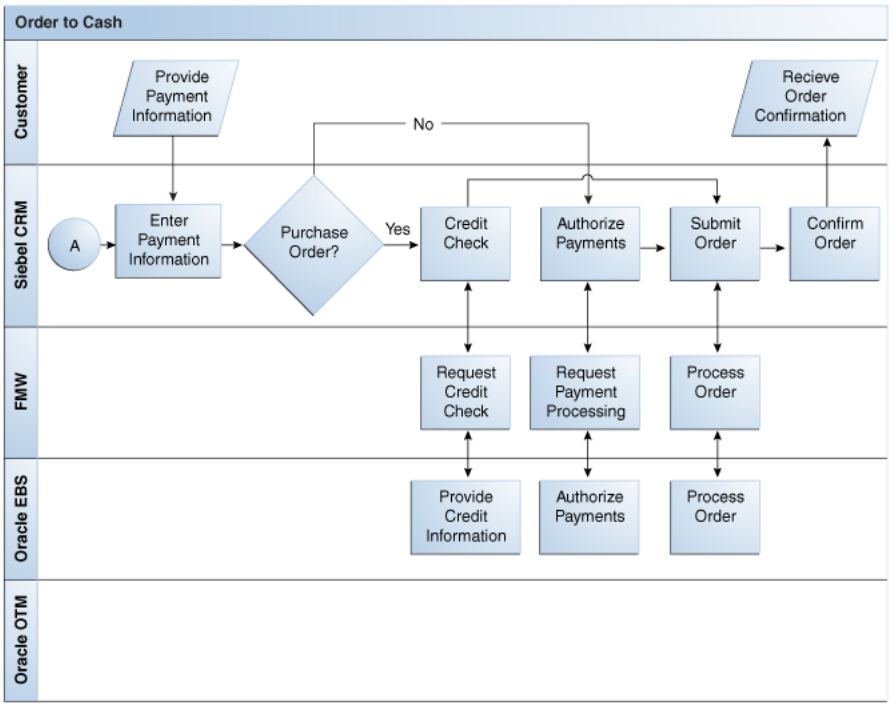


Figure 7–2 ATP Check Integration Flow (2 of 2)



7.1.1 Prerequisites

Find prerequisites in Section 13.6, "Available to Promise Check: Prerequisites and Data Requirements."

7.1.2 Solution Assumptions and Constraints

These are the assumptions for the ATP Check integration flow:

1. Support for ATP Check is only for simple products (that is, Items).
2. ATP Check cannot be performed against a complex product structure, namely pick-to-order (PTO) or assemble-to-order (ATO).
3. Available quantity at the line-level appears only when the status is *Available*.

   For any other ATP status, the quantity value at the line-level is not supplied. If the status is *Out of Stock*, *ATP not applicable*, or *Plan not found*, then the available quantity shown at the line-level is omitted and the actual available status appears at the Promise line-level.
4. The default requested date that appears in the Siebel CRM fulfillment view is the system date plus one day.

   This value is used for the ATP Check, and the resultant availability reflects this date.

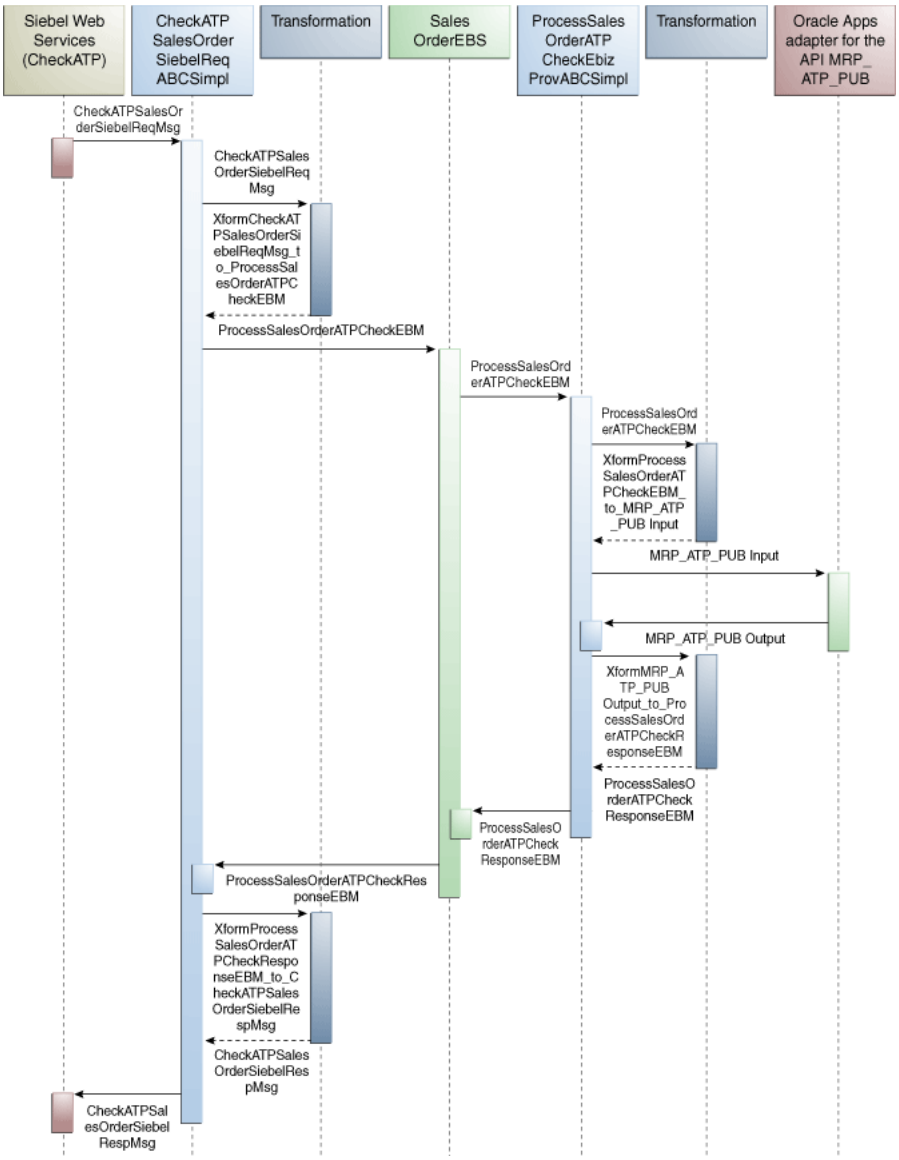
7.1.3 ATP Check Integration Flow

This integration flow uses the following interfaces:

- CheckATPSalesOrderSiebelReqABCSImpl
- SalesOrderEBS
- ProcessSalesOrderATPCheckEbizProvABCSImpl

Figure 7–3 illustrates the ATP Check integration scenario.

Figure 7–3 ATP Check Flow Sequence Diagram



1. From the Siebel Order window, search and select your order.
   a. In the Order window, select the Fulfillment tab.

      A line item appears on the line item applet.

   b. Navigate to the line item applet.

      The promise schedule line applet and request schedule line applet appear at the bottom of the window.

   c. Select a single line item, and click Inquire on the line item applet to check the availability of the current line item.

      Alternatively, click Inquire All to check the availability of all line items for the order.
2. Invoke the Siebel web service that calls the CheckATPSalesOrderSiebelReqABCSImpl with the operation CheckATP.

3. The CheckATPSalesOrderSiebelReqABCSImpl transforms the CheckATPSalesOrderReqMsg into the ProcessSalesOrderATPCheckEBM and invokes the SalesOrderEBS with the operation ProcessSalesOrderATPCheck.
4. The SalesOrderEBS routes the ProcessSalesOrderATPCheckEBM and the operation ProcessSalesOrderATPCheck routes it to the ProcessSalesOrderATPCheckEbizProvABCSImpl.
5. The ProcessSalesOrderATPCheckEbizProvABCSImpl, with the input message ProcessSalesOrderATPCheckEBM, does a transformation from ProcessSalesOrderATPCheckEBM to the input of the MSC_ATP_BPEL-24CALL_ATP_BPEL API.
6. The response of the MSC_ATP_BPEL-24CALL_ATP_BPEL API is transformed back to the ProcessSalesOrderATPCheckResponseEBM and the response is sent back to the SalesOrderEBS.
7. The SalesOrderEBS routes the ProcessSalesOrderATPCheckResponseEBM to the invoking CheckATPSalesOrderSiebelReqABCSImpl, using the ProcessSalesOrderATPCheck operation.
8. The CheckATPSalesOrderSiebelReqABCSImpl transforms the ProcessSalesOrderATPCheckResponseEBM into the CheckATPSalesOrderRespMsg and the response is sent back to the invoking Siebel web service.
9. If the integration flow is successful, the ATP amounts appear in Siebel CRM.

7.2 Siebel CRM Interfaces

For the ATP Check integration flow, these are the Siebel CRM web services:

- ATPCheckInterfaceRequestOrdersIO.xsd

  Siebel should call CheckATPSalesOrderSiebelReqABCSImpl with a CheckATPSalesOrderReqMsg ABM.

- ATPCheckInterfaceResponseOrdersIO.xsd

  Siebel expects CheckATPSalesOrderRespMsg ABM from CheckATPSalesOrderSiebelReqABCSImpl.

For more information about Siebel web services, see *CRM Web Services Reference*.

7.3 Oracle EBS Interfaces

For the ATP integration flow, these are the Oracle EBS web services:

- APPS_XX_BPEL_EBS11I10ATPCHECK_MSC_ATP_BPEL-24CALL_ATP_BPEL.xsd

  Service Name: XX_BPEL_EBS11I10ATPCHECK. MSC_ATP_BPEL$CALL_ATP_BPEL

- APPS_XX_BPEL_EBS11I10ATPCHECK_MSC_ATP_BPEL-24CALL_ATP_BPEL.xsd

  Service Name: XX_BPEL_EBS11I10ATPCHECK. MSC_ATP_BPEL$CALL_ATP_BPEL

For more information about Oracle E-Business Suite web services and documentation prior to Release 12.1.3, see the library on Oracle Technology Network: http://www.oracle.com/technetwork/documentation/applications-167706.html?. For Oracle E-Business Suite documentation for R12.1.3 and beyond, see this library: http://download.oracle.com/docs/cd/E18727_01/index.htm?

7.4 Core Oracle AIA Components

The ATP Check integration flow uses the following delivered horizontal components:

- SalesOrderEBO
- ProcessSalesOrderATPCheckEBM
- ProcessSalesOrderATPCheckResponseEBM
- SalesOrderEBS

The industry enterprise business object (EBO) and enterprise business message XML schema (EBM XSD) files are located here: $AIA_HOME/AIAMetaData/AIAComponents/EnterpriseObjectLibrary/Core/EBO/

The industry enterprise business service (EBS) WSDL files are located here: $AIA_HOME/AIAMetaData/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/

For detailed documentation of individual EBOs and EBMs, click the Oracle AIA Reference Doc link on EBO and EBM detail pages in Oracle Enterprise Repository.

For more information about using the Oracle Enterprise Repository and configuring it to provide the AIA Reference Doc link, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring and Using Oracle Enterprise Repository as the Oracle AIA SOA Repository."

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

For more information, see *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Extensibility."

7.5 Integration Services

The following services are delivered with the ATP Check integration flow:

- SalesOrderEBS
- CheckATPSalesOrderSiebelReqABCSImpl
- ProcessSalesOrderATPCheckEbizProvABCSImpl

For more information, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring and Using Oracle Enterprise Repository as the Oracle AIA SOA Repository."

7.5.1 SalesOrderEBS

For the ATP Check integration flow, the SalesOrderEBS exposes the QuerySalesOrderList operation and:

- Routes the ProcessSalesOrderATPCheckEBM to the Oracle EBS provider service.
- Routes the ProcessSalesOrderATPCheckResponseEBM to the requester service.

For more information about this EBS, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Designing and Developing Enterprise Business Services" and *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Enterprise Business Services."

7.5.2 CheckATPSalesOrderSiebelReqABCSImpl

The CheckATPSalesOrderSiebelReqABCSImpl is the application business connector service (ABCS) implementation that exposes the CheckATP operation related to ATP Check integration on the Siebel ABM.

The CheckATPSalesOrderSiebelReqABCSImpl transforms the Siebel Request Message into the ProcessSalesOrderATPCheckEBM and calls the SalesOrderEBS. The SalesOrderEBS routes it to the provider service ProcessSalesOrderATPCheckEbizProvABCSImpl. The provider service gets the ATPCheck details from the Oracle EBS and sends back the ProcessSalesOrderATPCheckResponseEBM to the routed EBS. The routing EBS then routes it to the CheckATPSalesOrderSiebelReqABCSImpl.

The CheckATPSalesOrderSiebelReqABCSImpl transforms the ProcessSalesOrderATPCheckResponseEBM to the Siebel response message and returns it to the calling Siebel web service.

The CheckATPSalesOrderSiebelReqABCSImpl has the following transformations:

- XformATPCheckInterfaceRequestOrders_to_ProcessSalesOrderATPCheckEBM
- XformProcessSalesOrderATPCheckResponseEBM_to_ ATPCheckInterfaceResponseOrders

7.5.3 ProcessSalesOrderATPCheckEbizProvABCSImpl

The ProcessSalesOrderATPCheckEbizProvABCSImpl BPEL process is used by the ProcessSalesOrderATPCheck integration flow. The SalesOrderEBS routes the ProcessSalesOrderATPCheckEBM message to the provider ABC ProcessSalesOrderATPCheckEbizProvABCSImpl. This is the provider implementation of ATP for Oracle EBS.

The ProcessSalesOrderATPCheckEbizProvABCSImpl transforms the ProcessSalesOrderATPCheckEBM message to an Oracle EBS API-specific message (ABM Message). The provider ABCS calls the Oracle EBS adapter with the ABM message and gets the ABM response message from Oracle EBS. The ProcessSalesOrderATPCheckEbizProvABCSImpl transforms this response ABM message to the ProcessSalesOrderATPCheckResponseEBM and sends it back to the SalesOrderEBS.

8

Shipping Charges Integration Flow

This chapter provides an overview of the process integration for Shipping Charges and discusses the shipping charges integration flow, Siebel Customer Relationship Management (Siebel CRM) and Oracle Transportation Management (OTM) interfaces, core Oracle Application Integration Architecture (Oracle AIA) components, and integration services.

This chapter includes the following sections:

8.1 Process Integration for Shipping Charges

The Shipping Charges integration flow can be called on request before calling the Order process integration flow. The Shipping Charges integration flow starts in Siebel CRM and obtains the estimated shipping cost from OTM. The source address, destination address, and weight information are passed to OTM to calculate the shipping charges. OTM responds with an amount, and this data is returned to Siebel CRM so that a Customer Service Representative (CSR) can inform the customer and continue with the order creation process.

The cost setup is maintained in OTM for all shipping locations (source and destination). The integration flow provides support for weight-based calculations only; the weight must be manually entered in Siebel CRM, and it is not included in the product synchronization integration flow.

The address, weight, and inventory source information are transferred to OTM to calculate and return the shipping charge amount. The Actual Freight charge calculations occur only at the time of shipping in Oracle E-Business Suite (Oracle EBS).

This integration flow works with Siebel sales orders only, not quotes.

Figure 8–1 and Figure 8–2 illustrate where the Shipping Charges integration flow fits in the Order to Cash integration flow.

Figure 8–1 Shipping Charges Integration Flow (1 of 2)

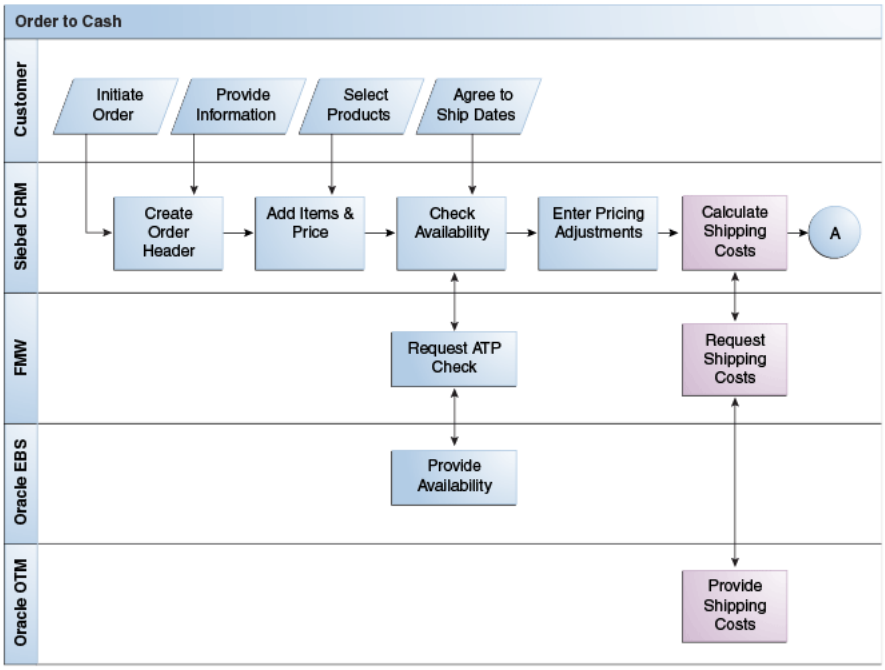
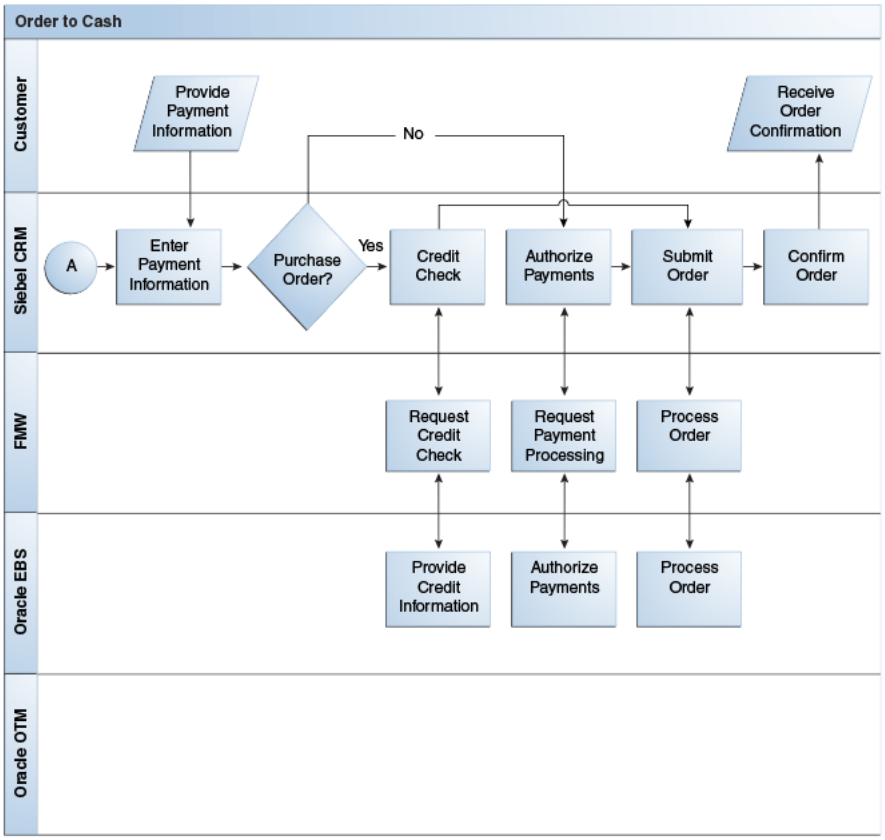


Figure 8–2 Shipping Charges Integration Flow (2 of 2)



8.1.1 Prerequisites

Find prerequisites in Section 13.7, "Shipping Charges: Prerequisites and Data Requirements."

8.1.2 Solution Assumptions and Constraints

The assumptions for the Shipping Charges integration flow are:

1. Shipping charges integration flow is not supported for complex products (such as Bill of Material (BoM)).
2. Shipping charges integration flow is not supported for Quotes.
3. Physical weight is the only criteria available for estimating shipping charges in this integration flow.

   The default value for the unit of measure (UOM) for physical weight is specified in the AIAConfigurationProperties.xml file.
4. The actual shipping charges are calculated in Oracle EBS when the order is fulfilled. These charges are synchronized to Siebel CRM in the Update Order flow.

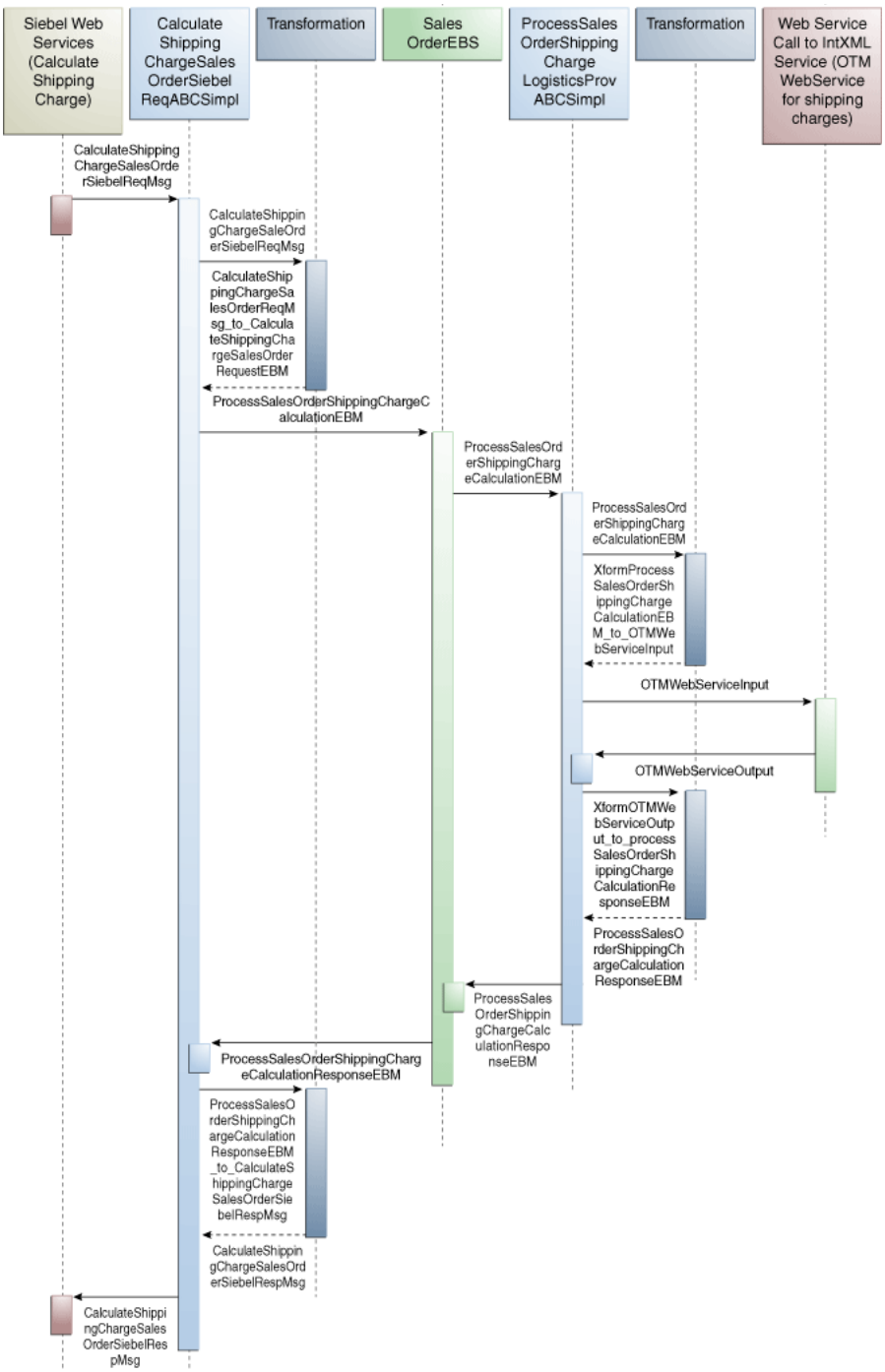
8.2 Shipping Charges Integration Flow

This integration flow uses the following interfaces:

- SalesOrderEBS
- CalculateShippingChargeSalesOrderSiebelReqABCSImpl
- ProcessSalesOrderShippingChargeLogisticsProvABCSImpl

Figure 8–3 illustrates the Shipping Charges integration scenario.

Figure 8–3 Shipping Charges Flow Sequence Diagram



1. In the Siebel Order window, search for and select your order.

   Order line items appear in the line item applet. Navigate to the Shipping tab, select the account, and ensure that it has a valid address; this is the ship-to address. Navigate to the Fulfillments subtab and select the Source at Header level or Source at Line level for all the line items. Navigate to the Summary tab and enter the weight of the product for all the simple products. Click Shipping Charges in the lower applet.

2. Invoke the Siebel web service, which calls the CalculateShippingChargeSiebelReqABCSImpl with the operation CalculateShippingCharge.
3. The CalculateShippingChargeSalesOrderSiebelReqABCSImpl transforms the CalculateShippingChargeSiebelReqMsg into the ProcessSalesOrderShippingChargeCalculationEBM and invokes the SalesOrderEBS with the operation ProcessSalesOrderShippingChargeCalculation and QueryCode ProcessSalesOrderShippingChargeCalculation.
4. Invoking the SalesOrderEBS with the message ProcessSalesOrderShippingChargeEBM and the operation CalculateShippingCharge and QueryCode ProcessSalesOrderShippingChargeCalculation routes the message to the ProcessSalesOrderShippingChargeLogisticsProvABCSImpl.
5. The ProcessSalesOrderShippingChargeLogisticsProvABCSImpl with input message ProcessSalesOrderShippingChargeCalculationEBM does a transformation to OTM web service input to invoke the OTM web service.
6. The OTM web service output is transformed into ProcessSalesOrderShippingChargeCalculationResponseEBM, and the response is returned to the SalesOrderEBS.
7. The SalesOrderEBS with the message ProcessSalesOrderShippingChargeCalculationResponseEBM and operation CalculateShippingCharge routes the message to the invoking ProcessSalesOrderShippingChargeLogisticsProvABCSImpl.
8. The CalculateShippingChargeSalesOrderSiebelReqABCSImpl transforms the ProcessSalesOrderShippingChargeCalculationResponseEBM to the CalculateShippingChargeSiebelRespMsg and sends the response back to the invoking sync web service.
9. The shipping charges are returned to and appear in Siebel CRM.

8.3 Siebel CRM Interfaces

The Siebel CRM interfaces for the Shipping Charges integration flow are:

- SWIOrderIO.xsd

  Siebel calls CalculateShippingChargeSalesOrderSiebelReqABCSImpl with a CalculateShippingChargeSalesOrderReqMsg application business message (ABM).

- SWIOrderIO.xsd

  Siebel expects a CalculateShippingChargeSalesOrderRespMsg ABM from CalculateShippingChargeSalesOrderSiebelReqABCSImpl.

For more information about Siebel web services, see *CRM Web Services Reference*.

8.4 OTM Interfaces

The OTM interfaces for the Shipping Charges integration flow are:

- GLOG.xsd

  Logis CalculateShippingCharge Request ABM

- GLOG.xsd

Logis CalculateShippingCharge Response ABM

For more information about OTM, see the product documentation

8.5 Core Oracle AIA Components

The Shipping Charges integration flow uses the following delivered horizontal components:

- SalesOrderEBO
- ProcessSalesOrderShippingChargeEBM
- ProcessSalesOrderShippingChargeResponseEBM

The industry enterprise business object (EBO) and enterprise business message XML schema (EBM XSD) files are located here: $AIA_HOME/AIAMetaData/AIAComponents/EnterpriseObjectLibrary/Core/EBO/

The industry enterprise business service (EBS) WSDL files are located here: $AIA_HOME/AIAMetaData/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/

For detailed documentation of individual EBOs and EBMs, click the AIA Reference Doc link on EBO and EBM detail pages in Oracle Enterprise Repository.

For more information about using the Oracle Enterprise Repository and configuring it to provide the AIA Reference Doc link, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring and Using Oracle Enterprise Repository as the Oracle AIA SOA Repository."

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

For more information, see *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Extensibility."

8.6 Integration Services

The following services are delivered with the Shipping Charges integration flow:

- SalesOrderEBS
- CalculateShippingChargeSalesOrderSiebelReqABCSImpl
- ProcessSalesOrderShippingChargeLogisticsProvABCSImpl

For more information, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring and Using Oracle Enterprise Repository as the Oracle AIA SOA Repository."

8.6.1 SalesOrderEBS

For the Shipping Charges integration flow, the SalesOrderEBS exposes the ProcessSalesOrderShippingChargeCalculation operation and:

- Routes ProcessSalesOrderShippingChargeCalculationEBM to the OTM provider service.
- Routes ProcessSalesOrderShippingChargeCalculationResponseEBM to the Requester service.

For more information about this EBS, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Designing and Developing Enterprise Business Services" and *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Enterprise Business Services."

8.6.2 CalculateShippingChargeSalesOrderSiebelReqABCSImpl

The CalculateShippingChargeSalesOrderSiebelReqABCSImpl is the application business connector service (ABCS) that exposes the following operations related to Shipping Charges integration on the Siebel ABM.

The ProcessSalesOrderShippingChargeCalculation integration flow uses this business process Execution Language (BPEL) process. The CalculateShippingChargeSalesOrderSiebelReqABCSImpl transforms the Siebel request message into the ProcessSalesOrderShippingChargeCalculationEBM and invokes the SalesOrderEBS with the operation ProcessSalesOrderShippingCharge.

The SalesOrderEBS routes the ProcessSalesOrderShippingChargeCalculationEBM to the provider service ProcessSalesOrderShippingChargeLogisticsProvABCSImpl. The provider service gets the ShippingCharge details from OTM and sends back the ProcessSalesOrderShippingChargeCalculationResponseEBM to the EBS. The EBS then routes this message to the CalculateShippingChargeSalesOrderSiebelReqABCSImpl.

The ProcessSalesOrderShippingChargeCalculation then transforms the ProcessSalesOrderShippingChargeCalculationResponseEBM to the Siebel response message and returns it to the calling Siebel web service.

The CalculateShippingChargeSiebelABCSImpl has the following transformations:

- XformShippingChargesSiebelReqMsg_to_ProcessSalesOrderShippingCharge CalculationEBM
- XformProcessSalesOrderShippingChargeCalculationResponseEBM_to_ ShippingCharges SiebelRespMsg

8.6.3 ProcessSalesOrderShippingChargeLogisticsProvABCSImpl

The ProcessSalesOrderShippingChargeLogisticsProvABCSImpl is part of the ProcessSalesOrderShippingChargeCalculation integration flow. This is the provider implementation for calculating shipping charges when OTM is used for logistics and fleet management.

The SalesOrderEBS routes the ProcessSalesOrderShippingChargeCalculationEBM message to the Provider ABCS ProcessSalesOrderShippingChargeLogisticsProvABCSImpl.

The ProcessSalesOrderShippingChargeLogisticsProvABCSImpl transforms the ProcessSalesOrderShippingChargeCalculationEBM message to the OTM web service message. The Provider ABCS calls the OTM web service and gets the response message from OTM. The ProcessSalesOrderShippingChargeLogisticsProvABCSImpl transforms this response ABM message to the ProcessSalesOrderShippingChargeCalculationResponseEBM and sends it back to the SalesOrderEBS.

9

Credit Check Integration Flow

This chapter provides an overview of Credit Check Requests and discusses Siebel Customer Relationship Management (Siebel CRM) and Oracle E-Business Suite (Oracle EBS) interfaces, core Oracle Application Integration Architecture (Oracle AIA) components, and integration services.

This chapter includes the following sections:

9.1 Credit Check Requests

If the order payment method is Purchase Order, then the Credit Check integration flow can be called on request before submitting the order. The Credit Check integration flow is a synchronous call, initiated in Siebel CRM that obtains the credit check status from Oracle EBS. The account Id and order amount are passed to Oracle EBS to check whether the order must be on credit hold. Oracle EBS responds with a message indicating whether to put the order on hold. This data is returned to Siebel CRM so that a Customer Service Representative (CSR) can inform the customer. The order is not sent to the back office until the hold is removed in Siebel CRM and the order is resubmitted.

If the credit check fails, then the Hold flag is set to *True* and the purchase order payment status is set to *Rejected*. When an order is submitted to Oracle EBS, if the Hold flag is enabled or if the purchase order status is not set to *Approved*, then the order is not submitted to the back office system for fulfillment.

This integration flow works with Siebel sales orders only, not quotes.

Figure 9–1 and Figure 9–2 illustrate where the Credit Check integration flow fits in the Order to Cash integration flow.

Figure 9–1 Credit Check Integration Flow (1 of 2)

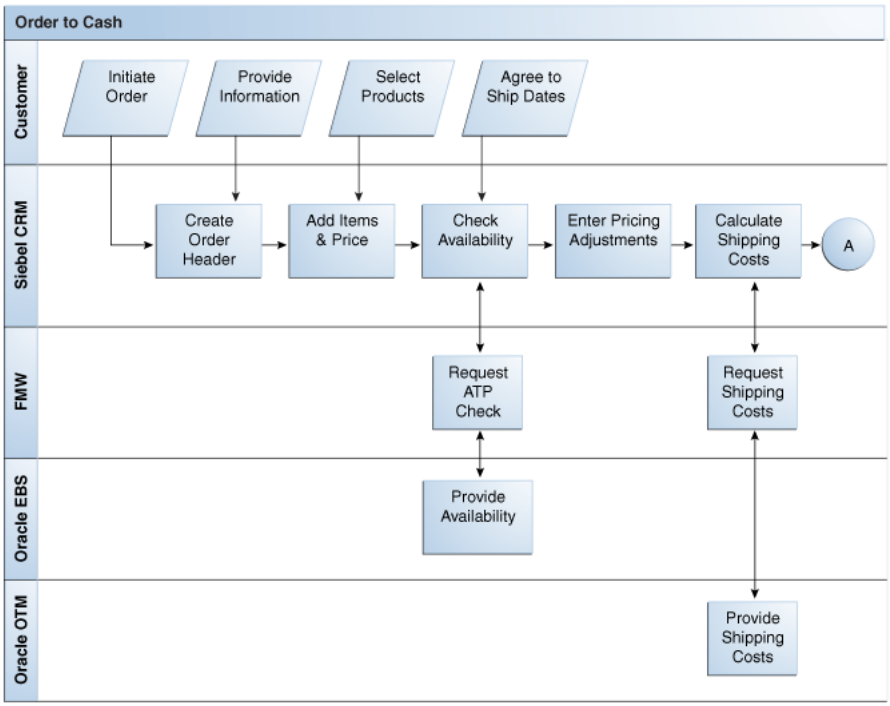
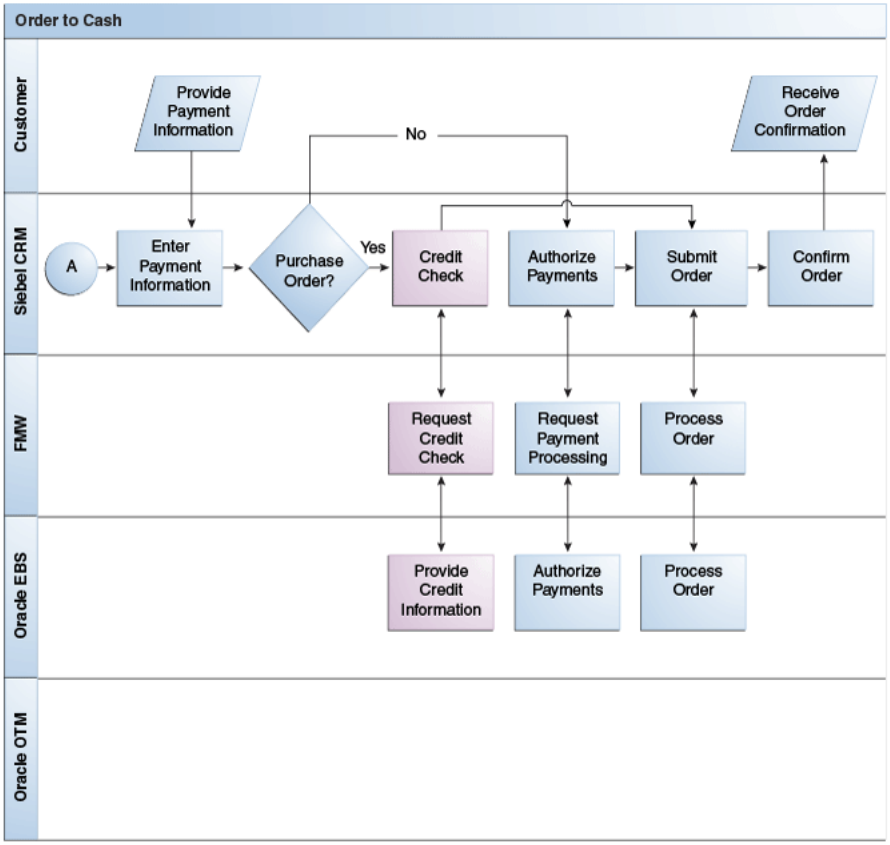


Figure 9–2 Credit Check Integration Flow (2 of 2)



9.1.1 Prerequisites

Find prerequisites in Section 13.8, "Credit Check: Prerequisites and Data Requirements."

9.1.2 Solution Assumptions and Constraints

These are the assumptions for the Credit Check integration flow:

1. The credit profile is maintained in Oracle EBS before credit check is invoked.
2. The order payment method is Purchase Order.
3. Credit check is not supported for Siebel quotes.

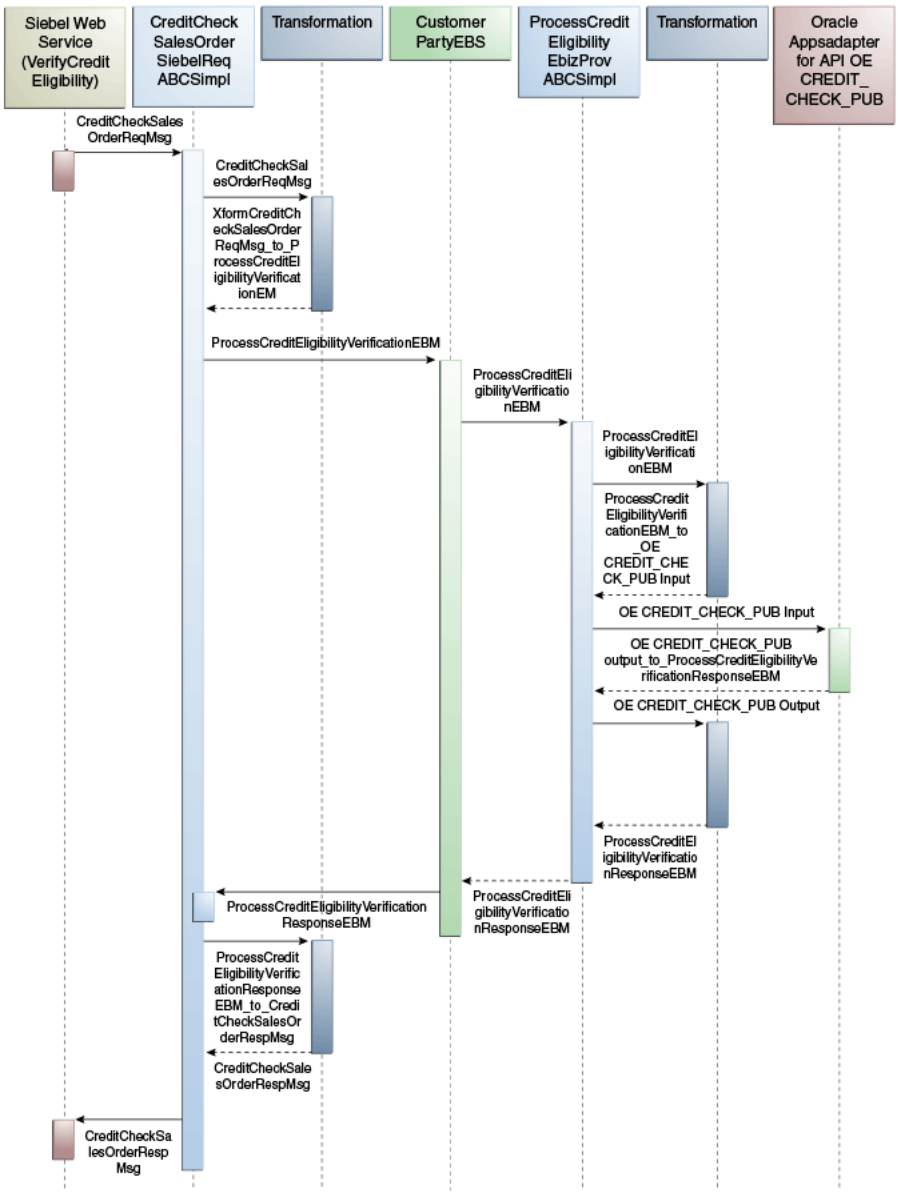
9.1.3 Credit Check Integration Flow

This integration flow uses the following interfaces:

- CustomerPartyEBS
- CreditCheckSalesOrderSiebelReqABCSImpl
- ProcessCreditEligibilityEbizProvABCSImpl

Figure 9–3 illustrates the Credit Check integration scenario.

Figure 9–3 Credit Check Flow Sequence Diagram



1. From the Siebel Order window, search for and select your order.

   Order line items appear in the line item applet. Navigate to the Payments applet and select Payment Lines. Click the New button on the Payment Lines tab. Enable the Payment Method option and select Purchase Order. Click outside the applet to save information.

   Navigate to the Payment Detail - Purchase Order applet. The Transaction Amount field equals the Transaction Amount field from the Payment Lines applet. The Credit Status, Credit Status As Of, and Credit Check Message fields are gray and set to null. Click Credit Check at the top of the applet. Customer credit information is returned to Siebel CRM from Oracle EBS.

2. The Siebel web service calls the CreditCheckSalesOrderSiebelReqABCSImpl with the operation CreditCheck.

3. The CreditCheckSalesOrderSiebelReqABCSImpl transforms the CreditCheckSiebleReqMsg into ProcessCreditEligibilityVerificationEBM and invokes the CustomerPartyEBS with the operation ProcessCreditEligibilityVerification.
4. Invoking the CustomerPartyEBS with the message ProcessCreditEligibilityVerificationEBM and the operation ProcessCreditEligibilityVerification routes the message to the ProcessCreditEligibilityEbizProvABCSImpl.
5. The ProcessCreditEligibilityEbizProvABCSImpl with the input message ProcessCreditEligibilityVerificationEBM does a transformation from ProcessCreditEligibilityVerificationEBM to the input of Oracle EBS OE CREDIT_CHECK_PUB API.
6. The response of the CREDIT_CHECK_PUB API is transformed back to the ProcessCreditEligibilityVerificationResponseEBM, and the response is sent back to the CustomerPartyEBS.
7. The CustomerPartyEBS with the message ProcessCreditEligibilityVerificationResponseEBM and operation ProcessCreditEligibilityVerification routes the message to the invoking CreditCheckSalesOrderSiebelReqABCSImpl.
8. The CreditCheckSalesOrderSiebelReqABCSImpl transforms the ProcessCreditEligibilityVerificationResponseEBM to the ProcessCreditEligibilityVerificationRespMsg, and the response is sent back to the invoking sync web service.
9. The result of the Credit Check appears in Siebel CRM. If the Credit Check is successful, the payment status field indicates *Approved* status.

9.2 Siebel CRM Interfaces

For the Credit Check integration flow, these are the Siebel CRM artifacts:

- SWIOrderIO.xsd

  Siebel calls the CreditCheckSalesOrderSiebelReqABCSImpl with a CreditCheckSalesOrderReqMsg application business message (ABM).

- SWIOrderIO.xsd

  Siebel expects the CreditCheckSalesOrderRespMsg ABM from CreditCheckSalesOrderSiebelReqABCSImpl.

For more information about Siebel web services, see *CRM Web Services Reference*.

9.3 Oracle EBS Interfaces

The Oracle EBS web services for the Credit Check integration flow are:

- APPS_OE_EXTERNAL_CREDIT_PUB_CHECK_EXTERNAL_CREDIT.xsd

  Service Name: OE_EXTERNAL_CREDIT_PUB. CHECK_EXTERNAL_CREDIT

- APPS_OE_EXTERNAL_CREDIT_PUB_CHECK_EXTERNAL_CREDIT.xsd

  Service Name: OE_EXTERNAL_CREDIT_PUB. CHECK_EXTERNAL_CREDIT

For more information about Oracle E-Business Suite web services and documentation prior to Release 12.1.3, see the library on Oracle Technology Network: http://www.oracle.com/technetwork/documentation/applications-167

706.html?. For Oracle E-Business Suite documentation for R12.1.3 and beyond, see this library: http://download.oracle.com/docs/cd/E18727_01/index.htm?

9.4 Core Oracle AIA Components

The Credit Check integration flow uses the following delivered horizontal components:

- CreditEligibility EBO
- ProcessCreditEligibilityVerificationEBM
- ProcessCreditEligibilityVerificationResponseEBM

The industry enterprise business object (EBO) and enterprise business message XML schema (EBM XSD) files are located here: $AIA_HOME/AIAMetaData/AIAComponents/EnterpriseObjectLibrary/Core/EBO/

The industry enterprise business service (EBS) WSDL files are located here: $AIA_HOME/AIAMetaData/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/

For detailed documentation of individual EBOs and EBMs, click the AIA Reference Doc link on EBO and EBM detail pages in Oracle Enterprise Repository.

For more information about using the Oracle Enterprise Repository and configuring it to provide the AIA Reference Doc link, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring and Using Oracle Enterprise Repository as the Oracle AIA SOA Repository."

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

For more information, see *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Extensibility."

9.5 Integration Services

The following services are delivered with the Credit Check integration flow:

- CustomerPartyEBS
- CreditCheckSalesOrderSiebelReqABCSImpl
- ProcessCreditEligibilityEbizProvABCSImpl

For more information, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring and Using Oracle Enterprise Repository as the Oracle AIA SOA Repository."

9.5.1 CustomerPartyEBS

For the Credit Check integration flow, the CustomerPartyEBS exposes the ProcessCreditEligibilityVerification operation related to the CustomerParty EBO and:

- Routes ProcessCreditEligibilityVerificationEBM to the Oracle EBS provider service
- Routes ProcessCreditEligibilityVerificationResponseEBM to the Siebel requester service

For more information about this EBS, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Designing and Developing

Enterprise Business Services" and *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Enterprise Business Services."

9.5.2 CreditCheckSalesOrderSiebelReqABCSImpl

The CreditCheckSalesOrderSiebelReqABCSImpl is the application business connector service (ABCS) that exposes the Credit Check operation related to the Credit Check integration on the Siebel ABM.

This Business process Execution Language (BPEL) process is used by the ProcessCreditEligibilityVerification integration flow. The CreditCheckSalesOrderSiebelReqABCSImpl transforms the Siebel Request message into the ProcessCreditEligibilityVerificationEBM and calls the CustomerPartyEBS. The CustomerPartyEBS routes it to the provider service ProcessCreditEligibilityEbizProvABCSImpl.

The provider service gets the CreditCheck details from Oracle EBS and sends back the ProcessCreditEligibilityVerificationResponseEBM to the routed EBS. The routing EBS then routes the message to the CreditCheckSalesOrderSiebelReqABCSImpl.

The CreditCheckSalesOrderSiebelReqABCSImpl then transforms the ProcessCreditEligibilityVerificationResponseEBM to the Siebel response message and returns it to the calling Siebel web service.

The CreditCheckSalesOrderSiebelReqABCSImpl has the following transformations:

- XformRequestCreditEligibilitySiebelReqMsg_to_ ProcessCreditEligibilityVerificationEBM
- XformProcessCreditEligibilityVerificationEBM_to_ CreditCheckSalesOrderRespMsg

9.5.3 ProcessCreditEligibilityEbizProvABCSImpl

The ProcessCreditEligibilityEbizProvABCSImpl BPEL process is used by the ProcessCreditEligibilityVerification integration flow. The CustomerPartyEBS routes the ProcessCreditEligibilityVerificationEBM message to the provider ABC ProcessCreditEligibilityEbizProvABCSImpl. This process is the provider implementation for credit eligibility verification when the application is Oracle EBS.

The ProcessCreditEligibilityEbizProvABCSImpl transforms the ProcessCreditEligibilityVerificationEBM message into an Oracle EBS application programming interface (API)-specific message (ABM message). The provider ABCS calls the Oracle EBS adapter with the ABM message and gets the ABM response message from Oracle EBS. The ProcessCreditEligibilityEbizProvABCSImpl transforms this response ABM message into the ProcessCreditEligibilityVerification ResponseEBM and sends it back to the CustomerPartyEBS.

10

Payment Authorization Integration Flow

This chapter provides an overview of Payment Authorization Requests and discusses Siebel Customer Relationship Management (Siebel CRM) and Oracle E-Business Suite (Oracle EBS) interfaces, core Oracle Application Integration Architecture (Oracle AIA) components, and integration services.

This chapter includes the following sections:

10.1 Payment Authorization Requests

If the payment method is Credit Card, then the Payment Authorization integration flow can be called on request before submitting the order.

The Payment Authorization integration flow is a synchronous call, initiated in Siebel CRM, which obtains the credit authorization status from Oracle EBS. The credit card details and order amount are passed to Oracle EBS to authorize the payment for the order amount. Oracle EBS responds with a message that indicates whether the amount was charged to the credit card or whether it failed authorization. This data is returned to Siebel CRM so that a Customer Service Representative (CSR) can inform the customer. The order is not sent to the back office until the payment authorization is successful.

Only one credit card is supported per order. If the credit authorization fails, the status is set to *Rejected*. An order cannot be submitted for fulfillment if the credit authorization fails.

This integration flow works with Siebel sales orders only, not quotes.

Note: The Payment Authorization flow applies to integration with Oracle EBS 11i10 only.

Figure 10–1 and Figure 10–2 illustrate where the Payment Authorization integration flow fits in the Order to Cash integration flow.

Figure 10–1 Payment Authorization Integration Flow (1 of 2)

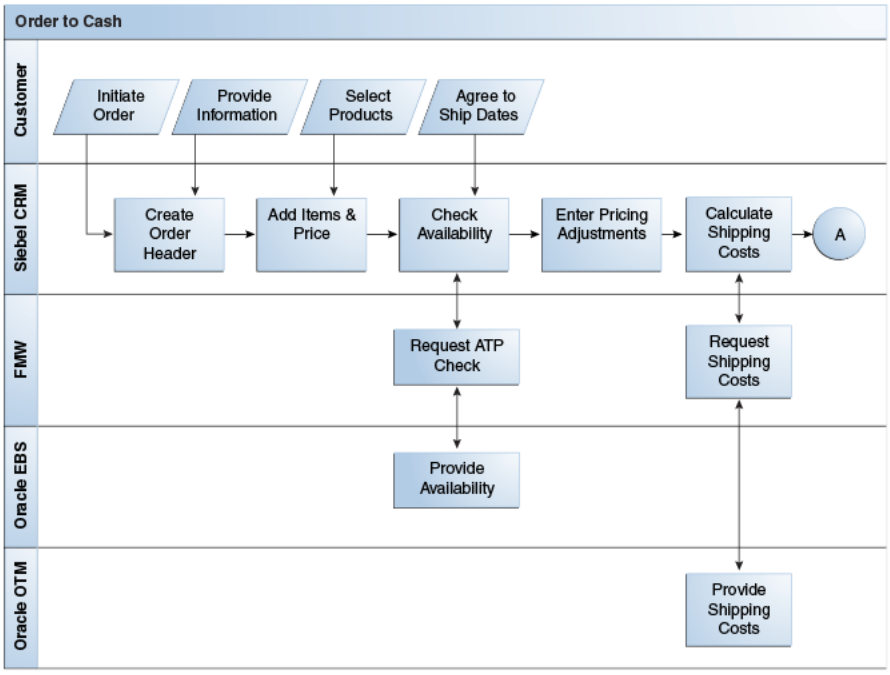
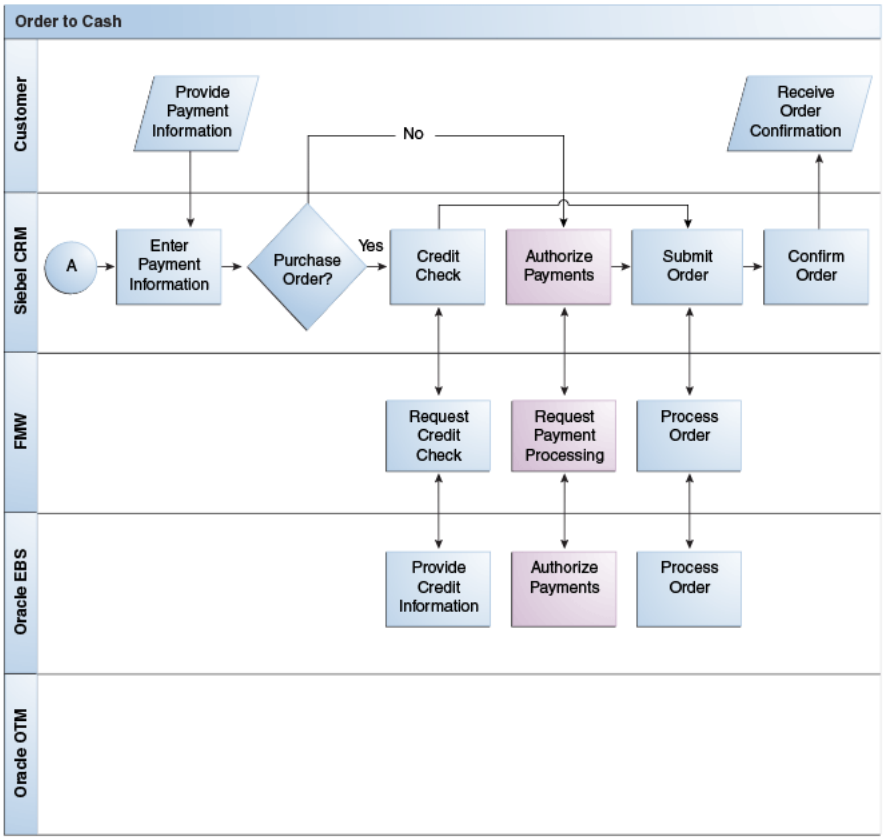


Figure 10–2 Payment Authorization Integration Flow (2 of 2)



10.1.1 Prerequisites

Find prerequisites in Section 13.9, "Payment Authorization: Prerequisites and Data Requirements."

10.1.2 Solution Assumptions and Constraints

The Payment Authorization integration flow has the following assumptions:

1. Only one credit card per order is supported.
2. Only Payment Authorization is supported. Settlement is not supported.
3. Payment Authorization is supported for sales orders only.
4. In this flow, the expiration month and year for the Credit Card Expiry details are passed from Siebel CRM.

   However, Oracle EBS also requires a day of the month. Therefore, the date format MM/YYYY from Siebel CRM is changed to YYYY-MM-DD format in Oracle EBS, where DD is set as 01 always to denote the first of the month. The issue with this approach is that the credit card is not authenticated in cases in which the current month, year, or both equals the expiry month, year, or both and the current date is beyond the first of the month.
5. This business flow of Payment Authorization integration is supported for Oracle EBS 11i10 only.

   This flow is not applicable for Oracle EBS 12.1.x due to inherent changes in the iPayment module. If implementations on Oracle EBS 12.1.x require Credit Card mode of payment, Oracle recommends following options:

   - Siebel CRM natively supports Credit Card authorization with third-party payment providers.

     The authorization codes received from this integration along with credit card details can be passed as part of the Order submit to Oracle EBS.

     For more information, see the *Siebel eSales Administration Guide*, "Integrating Siebel eSales with Third-Party Payment Applications. "
   - Alternately, only the credit card details can be synchronized to Oracle EBS as part of Order submit process, and the payment authorization can be accomplished in Oracle EBS itself.

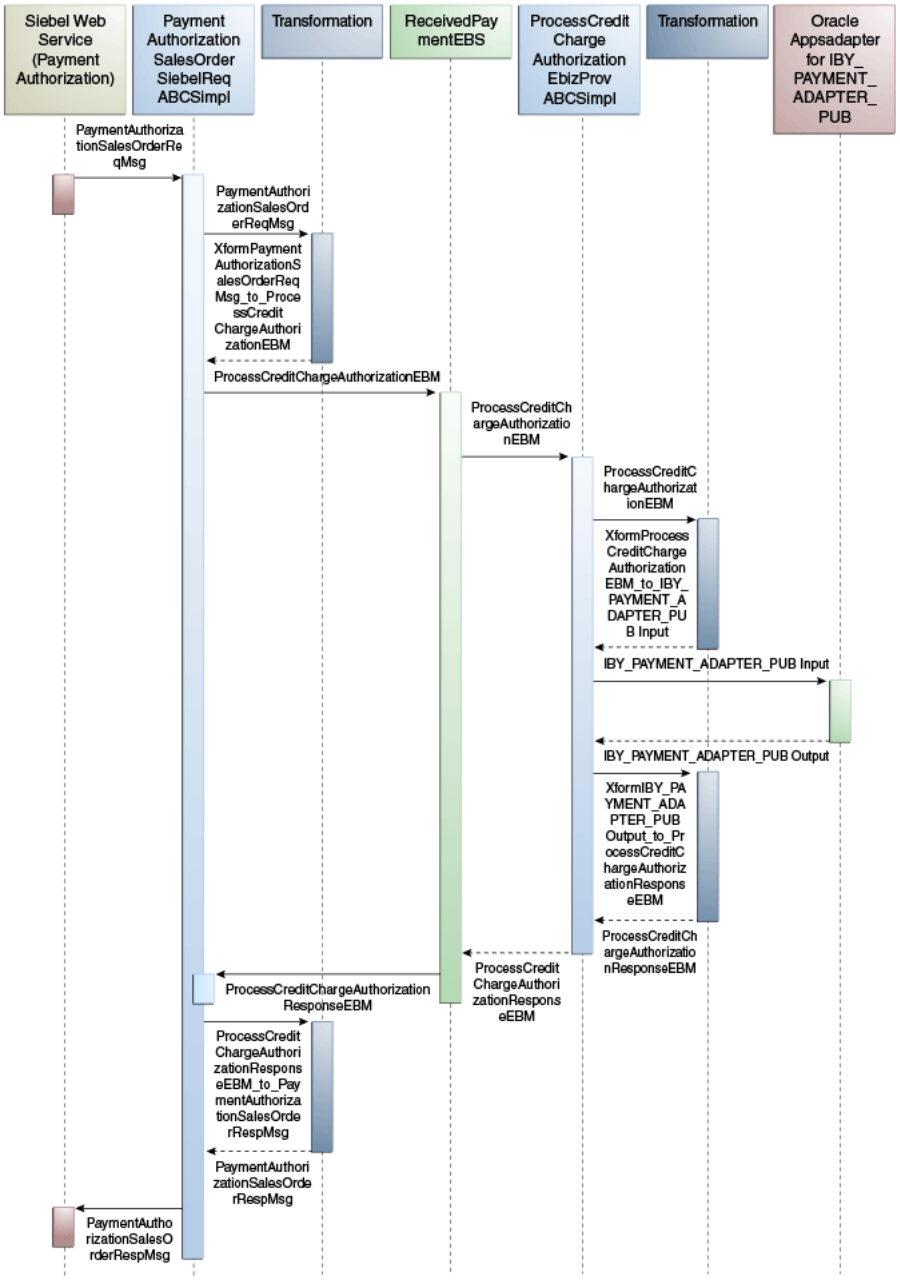
10.1.3 Payment Authorization Integration Flow

This integration flow uses the following interfaces:

- ReceivedPaymentEBS
- PaymentAuthorizationSalesOrderSiebelReqABCSImpl
- ProcessCreditChargeAuthorizationEbizProvABCSImpl

Figure 10–3 illustrates the Payment Authorization integration scenario.

Figure 10–3 Payment Authorization Flow Sequence Diagram



1. From the Siebel Order window, search for and select your order.

   Order line items appear in the line item applet. Navigate to the Siebel Payments window and enter specific payment attributes, such as credit card type, credit card number, credit card holder, and so on. Click Authorize.

2. Invoke the Siebel web service, which calls the PaymentAuthorizationSalesOrderSiebelReqABCSImpl with the operation PaymentAuthorization.

3. The PaymentAuthorizationSalesOrderSiebelReqABCSImpl transforms the PaymentAuthorizationSalesOrderReqMsg into ProcessCreditChargeAuthorizationEBM and invokes the ReceivedPaymentEBS with the operation ProcessCreditChargeAuthorization.

4. Invoking the ReceivedPaymentEBS with the message ProcessCreditChargeAuthorizationEBM and the operation ProcessCreditChargeAuthorization routes the message to the ProcessCreditChargeAuthorizationEbizProvABCSImpl.
5. The ProcessCreditChargeAuthorizationEbizProvABCSImpl with the input message ProcessCreditChargeAuthorizationEBM does a transformation from ProcessCreditChargeAuthorizationEBM to the input of the IBY_PAYMENT_ADAPTER_PUB API and calls the Oracle EBS-provided adapter for the procedure.
6. ProcessCreditChargeAuthorizationEbizProvABCSImpl transforms the output of the IBY_PAYMENT_ADAPTER_PUB API returned by the Oracle EBS adapter into the ProcessCreditChargeAuthorizationResponseEBM.

   The same EBM is returned to the ReceivedPaymentEBS.
7. The ReceivedPaymentEBS with the message ProcessCreditChargeAuthorizationResponseEBM and the operation ProcessCreditChargeAuthorization routes the message to the PaymentAuthorizationSalesOrderSiebelReqABCSImpl.
8. The PaymentAuthorizationSalesOrderSiebelReqABCSImpl transforms the ProcessCreditChargeAuthorizationResponseEBM into the PaymentAuthorizationSalesOrderRespMsg, and the response is sent back to the invoking web service.
9. If the integration flow is successful, the authorization code and date appear in Siebel CRM.

10.2 Siebel Customer Relationship Management (Siebel CRM) Interfaces

The Siebel interfaces for the Payment Authorization integration flow are:

- SWIOrderIO.xsd

  Siebel should call PaymentAuthorizationSalesOrderSiebelReqABCSImpl with a PaymentAuthorizationSalesOrderReqMsg application business message (ABM).
- SWIOrderIO.xsd

  Siebel expects PaymentAuthorizationSalesOrderRespMsg ABM from PaymentAuthorizationSalesOrderSiebelReqABCSImpl.

For more information about Siebel web services, see *CRM Web Services Reference*.

10.3 Oracle E-Business Suite (Oracle EBS) Interfaces

The Oracle EBS interfaces for the Payment Authorization integration flow are:

- APPS_IBY_BPEL_EBS11I10PAYMENTAUTH_IBY_PAYMENT_ADAPTER_PUB-24ORAPM.xsd

  Service Name: IBY_BPEL_EBS11I10PAYMENTAUTH IBY_PAYMENT_ADAPTER_PUB$ORAPM
- APPS_IBY_BPEL_EBS11I10PAYMENTAUTH_IBY_PAYMENT_ADAPTER_PUB-24ORAPM.xsd

  Service Name: IBY_BPEL_EBS11I10PAYMENTAUTH IBY_PAYMENT_ADAPTER_PUB$ORAPM

For more information about Oracle E-Business Suite web services and documentation prior to Release 12.1.3, see the library on Oracle Technology Network:

http://www.oracle.com/technetwork/documentation/applications-167706.html?. For Oracle E-Business Suite documentation for R12.1.3 and beyond, see this library: http://download.oracle.com/docs/cd/E18727_01/index.htm?

10.4 Core Oracle Application Integration Architecture (Oracle AIA) Components

The Payment Authorization integration flow uses the following horizontal components:

- ReceivedPaymentEBO
- ReceivedPaymentEBS
- ProcessCreditChargeAuthorizationEBM
- ProcessCreditChargeAuthorizationResponseEBM

The industry enterprise business object (EBO) and enterprise business message XML schema (EBM XSD) files are located here: $AIA_HOME/AIAMetaData/AIAComponents/EnterpriseObjectLibrary/Core/EBO/

The industry enterprise business service (EBS) WSDL files are located here: $AIA_HOME/AIAMetaData/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/

For detailed documentation of individual EBOs and EBMs, click the AIA Reference Doc link on EBO and EBM detail pages in Oracle Enterprise Repository.

For more information about using the Oracle Enterprise Repository and configuring it to provide the AIA Reference Doc link, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring and Using Oracle Enterprise Repository as the Oracle AIA SOA Repository."

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

For more information, see *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Extensibility."

10.5 Integration Services

The following services are delivered with the Payment Authorization integration flow:

- ReceivedPaymentEBS
- PaymentAuthorizationSalesOrderSiebelReqABCSImpl
- ProcessCreditChargeAuthorizationEbizProvABCSImpl

For more information, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring and Using Oracle Enterprise Repository as the Oracle AIA SOA Repository."

10.5.1 ReceivedPaymentEBS

For the Payment Authorization integration flow, the ReceivedPaymentEBS exposes the ProcessCreditChargeAuthorization operation and:

- Routes the ProcessCreditChargeAuthorizationEBM to the Oracle EBS provider service

- Routes ProcessCreditChargeAuthorizationResponseEBM to the requester service

For more information about this service, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Designing and Developing Enterprise Business Services" and *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Enterprise Business Services."

10.5.2 PaymentAuthorizationSalesOrderSiebelReqABCSImpl

The PaymentAuthorizationSalesOrderSiebelReqABCSImpl is the application business connector service (ABCS) that exposes the following operations related to Payment Authorization integration on the Siebel ABM.

This Business Process Execution Language (BPEL) process is used by the ProcessCreditChargeAuthorization integration flow. The PaymentAuthorizationSalesOrderSiebelReqABCSImpl transforms the Siebel request message into the ProcessCreditChargeAuthorizationEBM and calls the ReceivedPaymentEBS. The ReceivedPaymentEBS routes the message to the provider service ProcessCreditChargeAuthorizationEbizProvABCSImpl. The provider service sends a request for payment authorization to iPayment and gets the payment authorization details from Oracle EBS and sends back ProcessCreditChargeAuthorizationResponseEBM to the ReceivedPaymentEBS EBS. The ReceivedPaymentEBS then routes ProcessCreditChargeAuthorizationResponseEBM to the PaymentAuthorizationSalesOrderSiebelReqABCSImpl.

The PaymentAuthorizationSalesOrderSiebelReqABCSImpl then transforms the ProcessCreditChargeAuthorizationResponseEBM into the Siebel response message and returns it to the calling Siebel web service ProcessCreditChargeAuthorizationEbizProvABCSImpl.

PaymentAuthorizationSalesOrderSiebelReqABCSImpl has the following transformations:

- Xform_ListOfSWIOrderIO_to_ProcessCreditChargeAuthorizationEBM
- Xform_ProcessCreditChargeAuthorizationResponeEBM_to_ListOfSWIOrderIO

10.5.3 ProcessCreditChargeAuthorizationEbizProvABCSImpl

The ProcessCreditChargeAuthorizationEbizProvABCSImpl BPEL process is used by the ProcessCreditChargeAuthorization integration flow. The ReceivedPaymentEBS routes the ProcessCreditChargeAuthorizationEBM message to the provider ABCS ProcessCreditChargeAuthorizationEbizProvABCSImpl.

The ProcessCreditChargeAuthorization EbizProvABCSImpl transforms the ProcessCreditChargeAuthorizationEBM message to the Oracle EBS API-specific ABM message. The Provider ABCS calls the Oracle EBS adapter with the ABM message and gets the ABM response message from Oracle EBS. The ProcessCreditChargeAuthorizationEbizProvABCSImpl transforms this response ABM message into the ProcessCreditChargeAuthorizationResponseEBM and sends it back to the ReceivedPaymentEBS.

The ProcessCreditChargeAuthorizationEbizProvABCSImpl has the following transformations:

- ProcessCreditChargeAuthorizationEBM_to_IBY_PAYMENT_ADAPTER_PUB-24ORAPM_InputParameters

- IBY_PAYMENT_ADAPTER_PUB-24ORAPM_OutputParameters_to_ProcessCreditChargeAuthorizationResponseEBM

11

Process Integration for Order Management

This chapter provides an overview of the process integration for Order Management and discusses how to create, update, revise, and cancel sales orders. Also discussed are Siebel Customer Relationship Management (Siebel CRM) and Oracle E-Business Suite (Oracle EBS) interfaces, core Oracle Application Integration Architecture (Oracle AIA) components, and integration services.

This chapter includes the following sections:

11.1 Process Integration for Order Management

The process integration for Order Management between Siebel CRM and Oracle EBS supports the following integration flows:

- **Create sales orders:** This flow enables the processing of new orders submitted from Siebel CRM to Oracle EBS.
- **Revise sales order** (Siebel CRM-initiated): This flow enables the processing of order revisions or changes from Siebel CRM to Oracle EBS.
- **Cancel sales orders:** This flow enables the processing of order cancellations from Siebel CRM to Oracle EBS.
- **Update sales orders** (Oracle EBS-initiated): This flow enables the synchronization of order updates from Oracle EBS to Siebel CRM. Shipping details from Oracle EBS are also synchronized.

  Additionally, freight charges and estimated taxes are synchronized from Oracle EBS to Siebel CRM.

The process integration for Order Management uses the following service calls:

- **Available to Promise (ATP):** This check service is initiated from Siebel CRM and provides information about when a product can be fulfilled from Oracle EBS.
- **Credit Check:** This service is initiated from Siebel CRM and obtains credit status information for purchase orders from Oracle EBS.
- **Payment Authorization:** This service is initiated in Siebel CRM and obtains the authorization status for credit cards from iPayment in Oracle EBS.
- **Shipping Charges:** This service is initiated in Siebel CRM and obtains the estimated shipping cost from Oracle Transportation Management (OTM).

11.1.1 Prerequisites

Find prerequisites in Section 13.10, "Order Management: Prerequisites and Data Requirements."

11.1.2 Solution Assumptions and Constraints

The process integration for Order Management has the following assumptions and constraints:

1. Orders must be submitted with a status of *Booked*. Orders have a status of *Booked* before they are submitted. Before invoking any integration services, Siebel CRM validates this status using a workflow process.
2. Siebel CRM allows revisions to an order if the order status is *Booked*.
3. Business-to-Customer (B2C) scenarios are not supported.
4. Order updates in Oracle EBS are system-driven. When any of the following attributes change, they are synchronized to Siebel CRM:
   - Order Header/Line Status
   - Order Header Hold/Hold Release
   - System-generated line splits due to partial shipping
   - Line schedule arrival date changes
   - Taxes and freight charges
   - Shipping charges
5. The following Order (or Order line) statuses from Oracle EBS are supported:
   - Booked
   - Awaiting Shipping
   - Shipped
   - Fulfilled
   - Closed
   - Cancelled
   - Supply Eligible

   Additional status values can configured by exposing them in Oracle EBS, and adding List of Values (LOV) in Siebel, and domain value maps (DVMs) in Oracle AIA.

6. Customer Account Id, Product Id, and Price List Id (optional) must be available before an order is submitted. If they are not synchronized, the processes end in error.
7. When an order with a configured product is submitted to Oracle EBS, the order gets updated with certain included items, products, or both associated with the configured product.

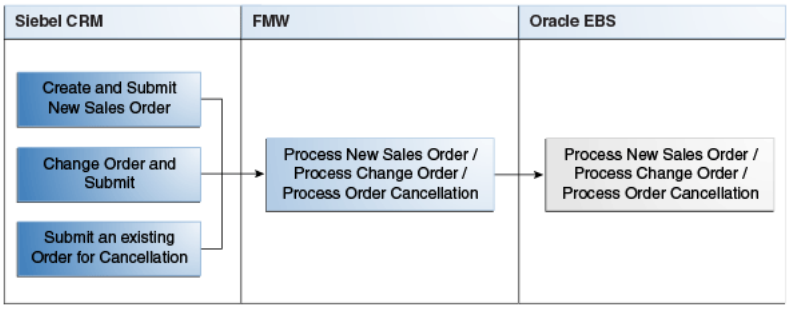
   This order update is synchronized back to Siebel CRM. However, these included items in Siebel CRM are displayed as separate line items, instead of being a part of configured product hierarchy. These included items do not have a pricing impact.
8. When an order is put on hold due to credit status, orders can be resubmitted only after the credit hold is released.
9. In Oracle EBS multiple holds can exist at the order header and line items. In this solution, only a single header hold is supported out-of-the-box.
10. Order data validation is accomplished using Siebel Data Validation Manager workflow processes. Order data is validated for a new order, order revision, and order cancellation flows in Siebel CRM.

    It is assumed that order data validation processes have been successfully run before submission of the order.
11. Order validation is not done within the integration process services.
12. This integration does not support manual updates to orders in Oracle EBS.
13. Only Siebel orders of the type *Sales Orders* are supported in this integration.
14. The order can be revised if the status of the order is *Booked* and the status of at least one order lines is *Booked* or *Awaiting Shipping*.
15. If the status of the line is *Shipped*, *Fulfilled*, *Closed*, *Cancelled*, *Cancel Pending*, or *Supply Eligible*, the order line is not revisable and cannot be canceled.
16. Deletion of an existing Order line in an order revision is not supported.
17. Return Material Authorization (RMA) is not supported for assemble-to-order (ATO) and pick-to-order (PTO) items.

11.1.3 Order Management Integration Flow

Figure 11–1 illustrates the overall flow for the process integration.

Figure 11–1 Overall Order Management Integration Flow



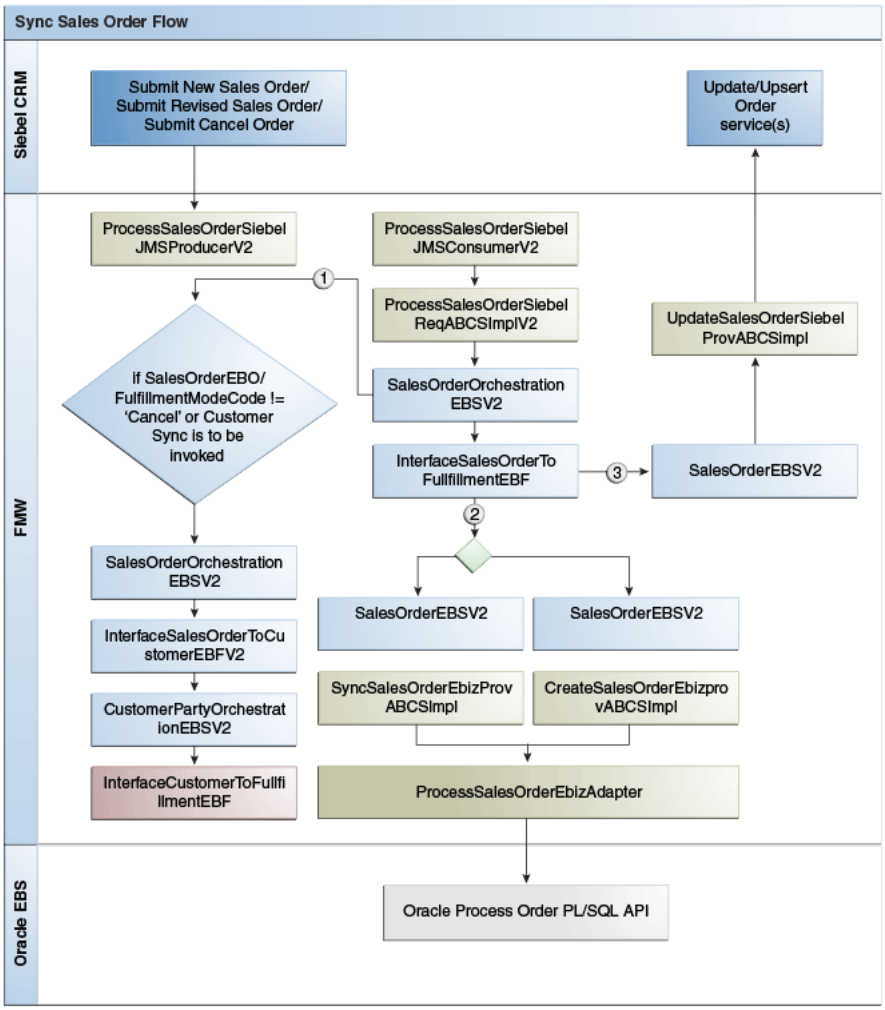
When this process is initiated, the following events take place:

1. Sales and RMA Orders are created in Siebel CRM.

2. Upon submission of the order in Siebel CRM, the Oracle AIA services send it to Oracle EBS for fulfillment.
3. Accounts created in Siebel CRM do not get immediately synchronized to Oracle EBS by default. Instead, accounts are synchronized to Oracle EBS only after an order has been submitted in Siebel CRM.
4. When an order is submitted in Siebel CRM, it raises an event that invokes the `ProcessSalesOrderSiebelReqABCSImplV2` service, which calls the `SalesOrderOrchestrationEBSV2` service. `SalesOrderOrchestrationEBSV2` can be configured to call an order orchestration process.
5. The sales order orchestration has two steps: first, `SyncCustomer` initiates the customer flows; second, `OrderFulfill`, which proceeds with order processing in Oracle EBS.
6. Automated order update events occur in Oracle EBS.
7. When a Sales or RMA order update event occurs in Oracle EBS, Oracle AIA services sends the order update to Siebel CRM.

Figure 11–2 illustrates the overall Order Management integration flow.

Figure 11–2 Order Management Overall Integration Flow



11.2 Sales Order Creation

The Create Sales Order flow enables the submission of new orders from Siebel CRM to Oracle EBS. Orders are captured in Siebel CRM. After orders go through the ATP check and credit check, they are submitted to Oracle EBS. After sales orders are submitted, they are frozen in Siebel CRM. When the order has been successfully submitted to Oracle EBS, the order status is updated in Siebel CRM.

The order submission initiates the synchronization of accounts from Siebel CRM to Oracle EBS.

These integration flows can be invoked before the Create Order flow is called:

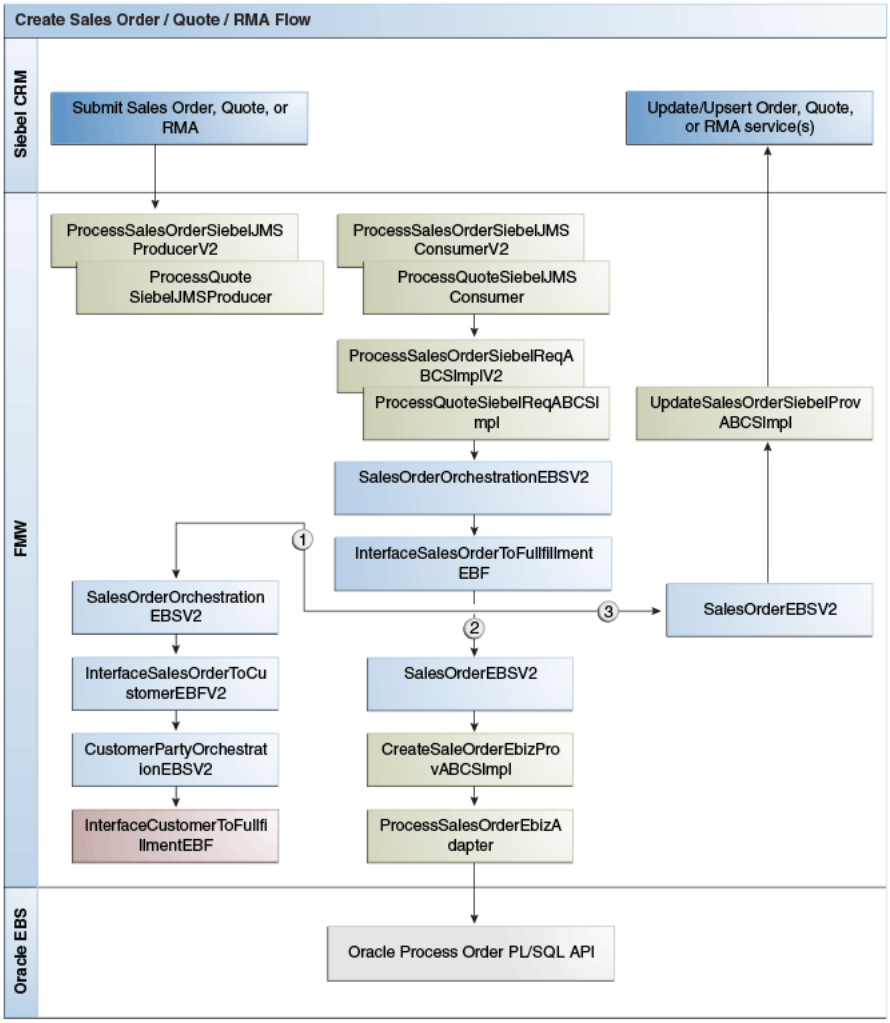
- ATP Check
- Shipping Charges
- Credit Check
- Payment Authorization

When a sales order is processed in Oracle EBS, the freight charges and estimated taxes are returned. The update sales order flow updates these on the Sales Order in Siebel CRM.

Additionally, when the order is fulfilled in Oracle EBS, the shipping details are synchronized to Siebel CRM.

Figure 11–3 illustrates the Create Sales Order integration flow.

Figure 11–3 Create Sales Order Integration Flow



11.2.1 Create Sales Order Integration Flow

This integration flow uses the following interfaces:

- ProcessSalesOrderSiebelJMSProducerV2
- ProcessSalesOrderSiebelJMSConsumerV2
- ProcessSalesOrderSiebelReqABCSImplV2
- SalesOrderOrchestrationEBSV2
- SalesOrderOrchestrationResponseEBSV2
- InterfaceSalesOrderToFulfillmentEBF
- InterfaceSalesOrderToCustomerEBF
- InterfaceCustomerToFulfillmentEBF
- SalesOrderEBSV2
- CreateSalesOrderEbizProvABCSImpl
- SyncSalesOrderEbizProvABCSImpl

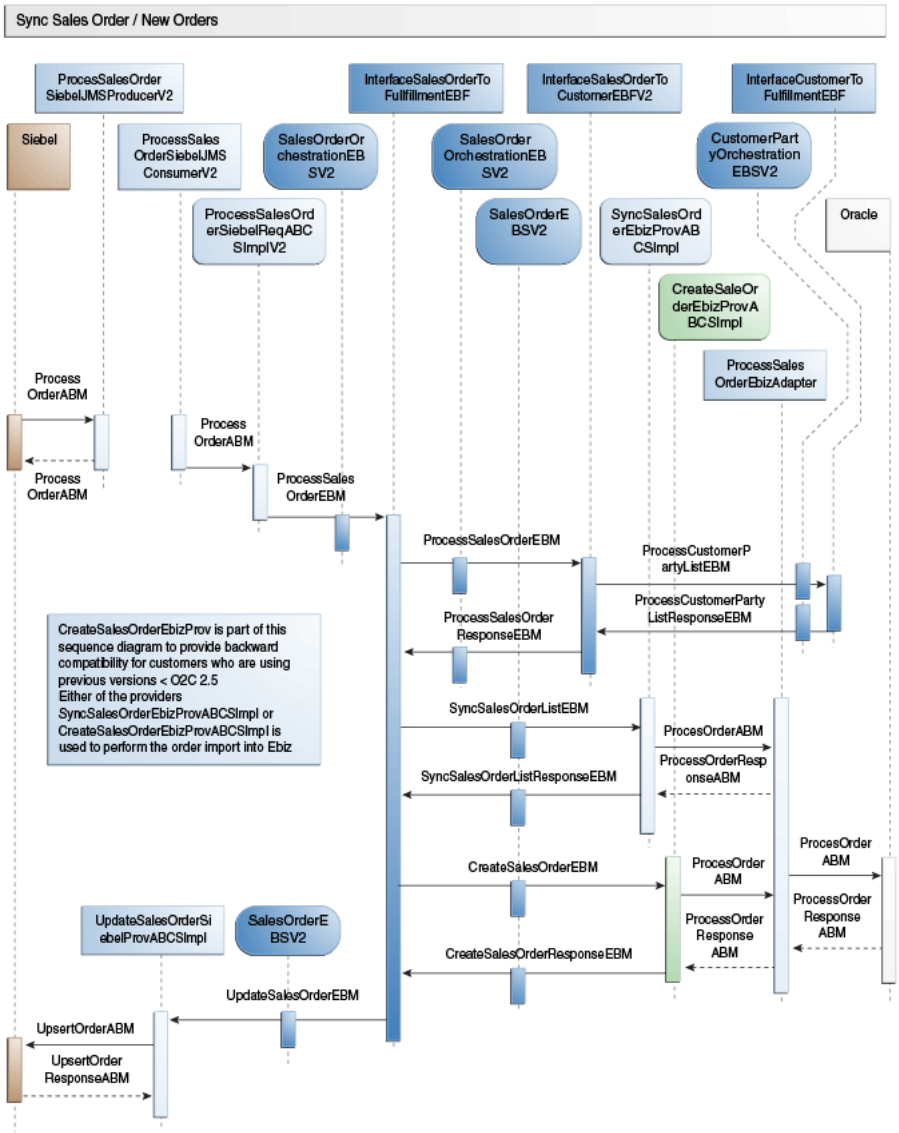
- ProcessSalesOrderEbizAdapter
- SalesOrderResponseEBSV2
- UpdateSalesOrderSiebelProvABCSImpl

Order components that are synchronized and passed to Oracle EBS:

- Accounts
- Addresses
- Contacts
- Products
- Price
- Order Management-related

Figure 11–4 illustrates the Create Sales Order integration scenario.

Figure 11–4 Create Sales Order Flow Sequence Diagram



When the Create Sales Order process is initiated, the following events occur:

1. When an Order is created and submitted in Siebel CRM (version 8.0.0.x), workflow invokes the ProcessSalesOrderSiebelJMSProducerV2 service. If Siebel CRM 8.1.1.x is being used, then Siebel directly enqueues the message onto the AIA queue.
2. The invoked JMSProducer service enqueues the Siebel application business message (ABM) to a Java Message Service (JMS) queue and replies the success or failure of the enqueue to the Siebel application.
3. The ProcessSalesOrderSiebelJMSConsumerV2 service dequeues the Siebel ABM from the JMS queue and invokes the ProcessSalesOrderSiebelReqABCSImplV2 service. In case of Siebel 8.1.1.x, the dequeue is done by ProcessSalesOrderSoapMsgSiebelJMSConsumer.
4. The invoked ProcessSalesOrderSiebelReqABCSImplV2 service transforms the Siebel ABM into the ProcessSalesOrderEBM.

   In doing so, the Order, Order Line, Account, Contact, and Address cross-reference tables are populated with the Siebel Row IDs and newly generated Common IDs. In this process of transformation, EBMHeader/VerbCode is identified as follows: If CurrentOrderRowId (SWIOrder/ID) equals PreviousOrderId (SWIOrder/PreviousOrderId), the VerbCode is populated as Process; otherwise, the VerbCode holds Sync.
5. The ProcessSalesOrderSiebelReqABCSImplV2 then invokes the ProcessSalesOrder operation of the SalesOrderOrchestrationEBSV2, which is routed to the InterfaceSalesOrderToFulfillmentEBF enterprise business flow.
6. The InterfaceSalesOrderToFulfillmentEBF process in turn invokes the InterfaceSalesOrderToCustomerEBF process only for the sales orders.

   A configuration parameter is available to prevent the customer sync service from being invoked if a customer chooses. However, if it is decided based on a flag sent by Siebel whether the Address or Contact has changed, then the Customer sync should be invoked.
7. The InterfaceSalesOrderToCustomerEBF process transforms the SalesOrderEBM into the ProcessCustomerPartyListEBM containing a reduplicated list of Account, Contact, and Address IDs that were referenced on the order.
8. The InterfaceSalesOrderToCustomerEBF then invokes the InterfaceCustomerToFulfillmentEBF, which performs the necessary steps for synchronizing the customer accounts, contacts, and addresses to the fulfillment system.
9. Upon receiving the response from the InterfaceCustomerToFulfillmentEBF service, the InterfaceSalesOrderToCustomerEBF sends a response enterprise business message (EBM) back to the InterfaceSalesOrderToFulfillmentEBF.
10. When the response is received from the InterfaceSalesOrderToCustomerEBF service, backward compatibility to support CreateSalesOrderEbizProvABCSImpl for new order creation can be controlled by a configuration parameter isLegacyEbizProvSupported and EBMHeader/VerbCode value Process (refer to step 4).

    The InterfaceSalesOrderToFulfillmentEBF performs a transformation to generate the CreateSalesOrderEBM, which is used to invoke the CreateSalesOrder operation of the SalesOrderEBSV2. Otherwise, the InterfaceSalesOrderToFulfillmentEBF performs a transformation to generate the SyncSalesOrderListEBM, which is used to invoke the SyncSalesOrderList operation of the SalesOrderEBSV2.

11. The SalesOrderEBSV2 routes the CreateSalesOrder invocation to the CreateSalesOrderEbizProvABCSImpl service or routes the SyncSalesOrderList invocation to the SyncSalesOrderEbizProvABCSImpl.
12. The CreateSalesOrderEbizProvABCSImpl/SyncSalesOrderEbizProvABCSImpl service transforms the CreateSalesOrderEBM/SyncSalesOrderListEBM into the Oracle ProcessOrderABM.
13. The CreateSalesOrderEbizProvABCSImpl/SyncSalesOrderEbizProvABCSImpl service then invokes the ProcessOrder operation of the ProcessSalesOrderEbizAdapter service.

    This service invokes the appropriate Oracle ProcessOrder PL/SQL application programming interface (API), which results in the creation of the order in the Oracle EBS system.
14. Upon completion and response from the ProcessSalesOrderEbizAdapter, the CreateSalesOrderEbizProvABCSImpl/SyncSalesOrderEbizProvABCSImpl generates the response EBM, during which the Oracle IDs are added to the cross-reference, and replies to the SalesOrderResponseEBSV2, which in turn is routed back to the InterfaceSalesOrderToFulfillmentEBF.
15. The InterfaceSalesOrderToFulfillmentEBF then performs another transformation to generate the UpdateSalesOrderEBM, which is used to invoke the UpdateSalesOrder operation of the SalesOrderEBSV2.
16. The SalesOrderEBSV2 routes the UpdateSalesOrder invocation to the UpdateSalesOrderSiebelProvABCSImpl service.
17. The UpdateSalesOrderSiebelProvABCSImpl transforms the UpdateSalesOrderEBM to the UpsertOrder Siebel ABM using the appropriate cross-reference tables to determine the Siebel IDs from the Common IDs.
18. The UpdateSalesOrderSiebelProvABCSImpl then invokes the Siebel UpsertOrder web service to update the status of the order header.

11.2.1.1 Target System Identification and Routing

An appropriate provider application business connector service (ABCS) should be identified and routed to the following two points in the Create Order integration flow:

- The creation of the order in the back office fulfillment system. The SalesOrderEBSV2 CreateSalesOrder operation must invoke the appropriate provider ABCS. When delivered, the target fulfillment system is not identified until the original CreateSalesOrderEBS routing rules are run and the system determines that the Oracle EBS provider ABCS should be the target. Customers can replace the original routing rules to do more complex target system decision-making and routing.
- After the creation of the order in the back office fulfillment when the SalesOrderEBSV2 UpdateSalesOrder operation must invoke the provider ABCS for the original CRM system from which the order was submitted. This routing uses the Source System Id that is available in the original ProcessSalesOrderEBM to identify the provider ABCS.

11.3 Sales Order Updates (Oracle EBS Initiated)

The Update Sales Order integration flow enables the synchronization of order updates from Oracle EBS to Siebel CRM. This is a one-way synchronization from Oracle EBS to Siebel CRM. When orders are updated, a business event is triggered to enable the

synchronization of the latest order status from Oracle EBS to Siebel CRM. Only the following order and order-line statuses are brought back to Siebel CRM:

- Booked
- Awaiting Shipping
- Shipped
- Fulfilled
- Closed
- Cancelled
- Supply Eligible

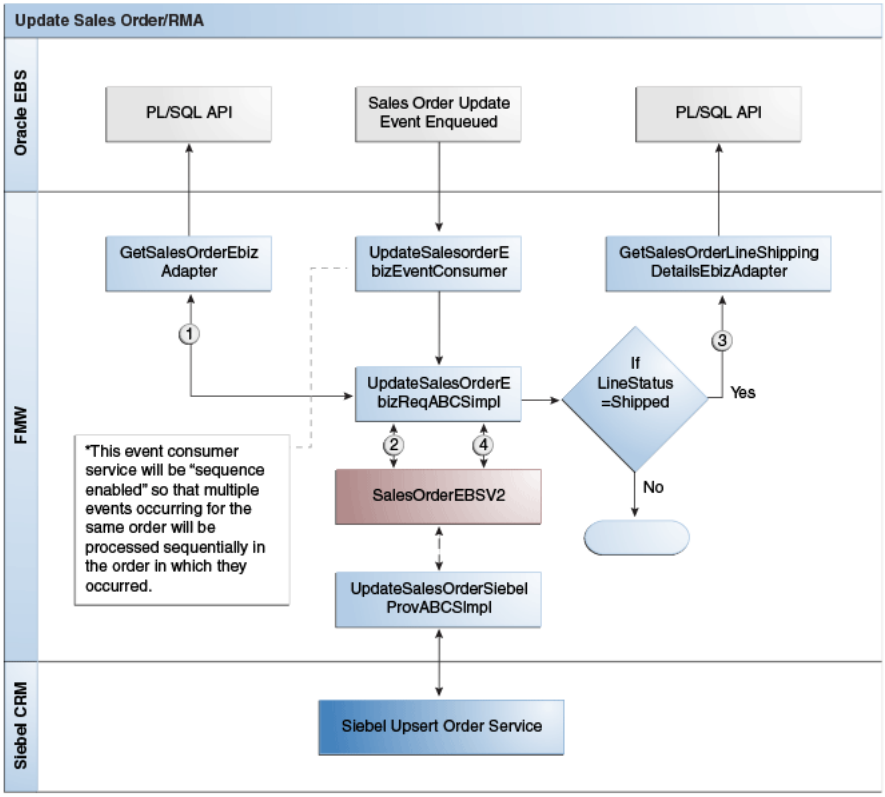
The Update Sales Order integration flow uses the same Sales Order enterprise business object (EBO) as the create process.

Certain Order updates in Oracle EBS are system-driven. When any of the following attributes changes, these are synchronized to Siebel CRM:

- Order Header/Line Status
- Order Header Hold/Hold Release
- System-generated line splits due to partial shipping
- Line schedule arrival date
- Line schedule ship date
- Order Header and Order Line Taxes
- Order Header and Order Line Freight charges
- When an order line is shipped, the following order line shipping details are also synchronized to Siebel CRM:
  - Tracking number
  - Actual shipped date
  - Ship from location
  - Carrier code
  - Shipped quantity

Figure 11–5 illustrates the Update Sales Order integration flow.

Figure 11–5 Update Sales Order Integration Flow

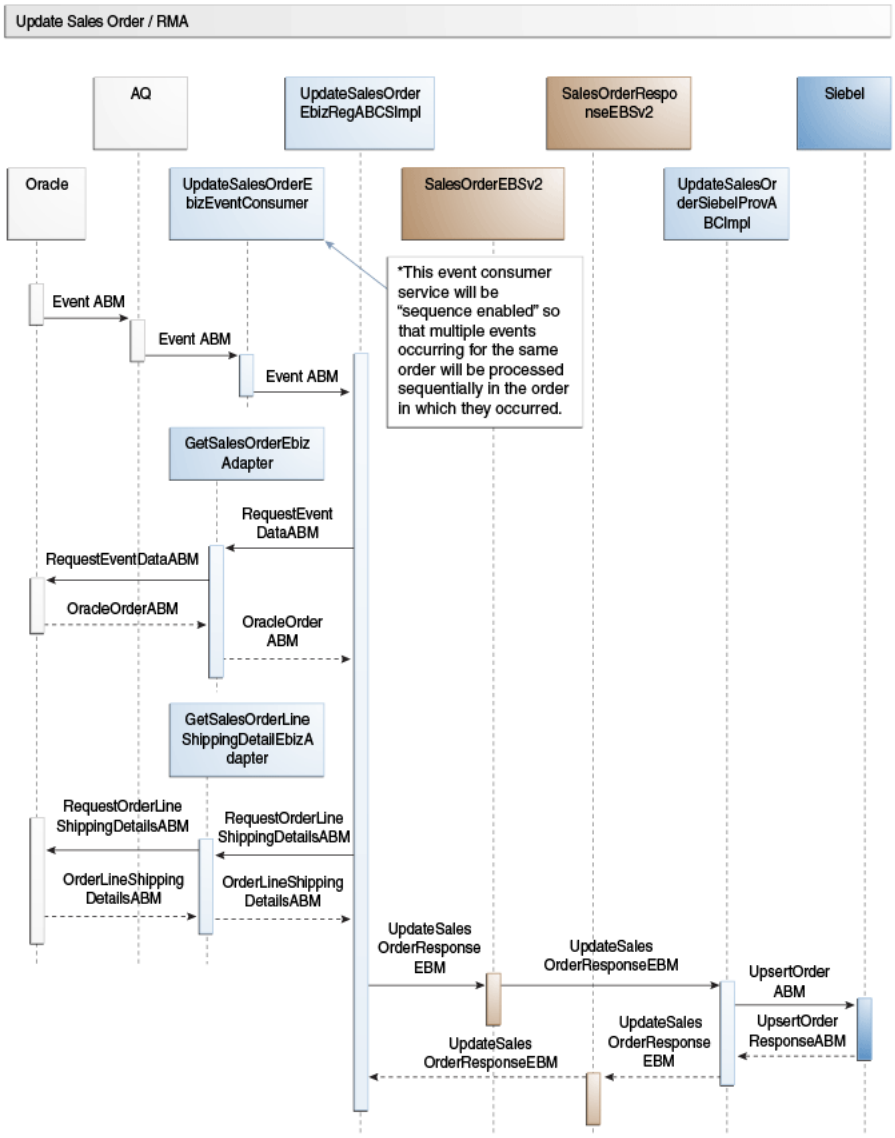


This integration flow uses the following interfaces:

- UpdateSalesOrderEbizEventConsumer
- UpdateSalesOrderEbizReqABCSImpl
- GetSalesOrderEbizAdpater
- SalesOrderEBSV2
- SalesOrderResponseEBSV2
- UpdateSalesOrderSiebelProvABCSImpl
- GetSalesOrderLineShippingDetailsEbizAdapter

Figure 11–6 illustrates the Update Sales Order integration scenario.

Figure 11–6 Update Sales Order Flow Sequence Diagram



When you initiate the Update Sales Order process in Oracle EBS, the following events take place:

1. When an order header or line update event is launched, the event is enqueued to Oracle Advanced Queuing (AQ).
2. UpdateSalesOrderEbizEventConsumer consumes the message from the AQ.

   The UpdateSalesOrderEbizEventConsumer service has resequencer properties defined to provide guaranteed first-in first-out processing of the events. For each event, the UpdateSalesOrderEbizReqABCSImpl service is invoked. If the resequencer is enabled, then events of the same order are processed sequentially but the events of a different order are processed in parallel.
3. The UpdateSalesOrderEbizReqABCSImpl service transforms the EventABM into the Oracle ABM for requesting the Order payload, and then invokes the GetSalesOrderEbizAdpater service for retrieving the full Order update details.

4. For each Order line with status Shipped, the Order payload is transformed into the RequestOrderLineShippingDetailsABM and the GetSalesOrderLineShippingDetailsEbizAdapter is invoked for retrieving shipping details for that shipped Order line.
5. The UpdateSalesOrderEbizReqABCSImpl then transforms the full Order update payload along with the fetched and aggregated OrderLineShippingDetailsABMs into the UpdateSalesOrderEBM to invoke the UpdateSalesOrder operation of the SalesOrderEBSV2.
6. The SalesOrderEBSV2 routes the UpdateSalesOrder operation to the UpdateSalesOrderSiebelProvABCSImpl service.
7. The UpdateSalesOrderSiebelProvABCSImpl service transforms the UpdateSalesOrderEBM into the Upsert Order Siebel ABM and invokes the Siebel Upsert Order web service.
8. Upon response from the Siebel web service, the response is transformed into the UpdateSalesOrderResponseEBM during which any newly inserted line IDs are added to the cross-reference.

In Oracle EBS, when the order is updated, a business event is triggered from Oracle EBS that enables the synchronization of the latest order status to Siebel CRM. These business events in Oracle EBS result in messages generated and sent to Oracle AIA for processing. Sequencing of events has been configured in the Update Order flow to provide the following functionality:

- Events belonging to different orders run in parallel.
- Events belonging to the same order run sequentially.

11.3.1 Target System Determination and Routing

The Update Order flow determines what target system provider ABC service to invoke. As delivered, the UpdateSalesOrderEBS routing rules run and determine that the Siebel provider ABC service should be the target. Customers can replace the original routing rules to do more complex target system decision-making and routing.

11.3.2 Updating Order Events Sequencing on Fusion Middleware

The following steps must be performed using the Oracle Enterprise Manager Fusion Middleware Control Console to enable the resequencer functionality in Oracle Mediator and for better performance. The Fusion Middleware (FMW) sequencing feature ensures better scalability and performance. Not defining these properties does not affect anything in the code, but the synchronize Update Order flow works as a single threaded flow.

To update order events resequencing on FMW:

1. Launch Oracle Enterprise Manager.
2. Open the SOA Infrastructure Home page.
3. From the **SOA Infrastructure** menu, select **SOA Administration** and then **Mediator Properties**.
4. Configure the following properties for better performance of the sequencing feature:
    - Resequencer Worker Threads = 5
    - Resequencer Locker Thread Sleep (sec) = 1000

- Resequencer Maximum Groups Locked = 100

Note: The value of these properties can vary based on the environment configuration and the same can be set appropriately.

5. Save the changes.

For more information about using the Oracle Mediator Resequencer as a part of an Oracle AIA implementation, see *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*, "Resequencing in Oracle Mediator."

11.4 Sales Order Revision (Siebel CRM Initiated)

All customer-requested revisions or changes to the order are done in Siebel CRM. If the order is booked and submitted to Oracle EBS, these changes must be validated by Oracle EBS Order Management processing constraints. The changes that are not allowed by the fulfillment system (Oracle EBS) are restricted in Siebel CRM, and the corresponding error message is sent

11.4.1 Solution Assumptions and Constraints

The process integration for Order Management has the following assumptions and constraints:

1. An order in Siebel CRM can be revised only when the status of the order is *Booked*.
2. An order line in Siebel CRM can be revised or canceled if its status is *Booked* or *Awaiting Shipping*. Siebel CRM does not allow order line revision if the status of the order line is *Shipped, Fulfilled, Closed, Cancelled, Cancel Pending*, or *Supply Eligible*. In this case, if the CSR clicks the **Revise** button, the message box appears with the message "Order not revise-able as it is beyond that state."
3. Siebel CRM controls the Order or Order line revision using rules in its Data Validation Manager component.
4. When a sales order is revised in Oracle EBS, the freight charges and estimated taxes are returned for those lines. The revise sales order flow updates these on the sales order in Siebel CRM.
5. Deletion of an existing line in an order in a revision is not supported.

11.4.2 Revise Sales Order (Siebel CRM Initiated) Integration Flow

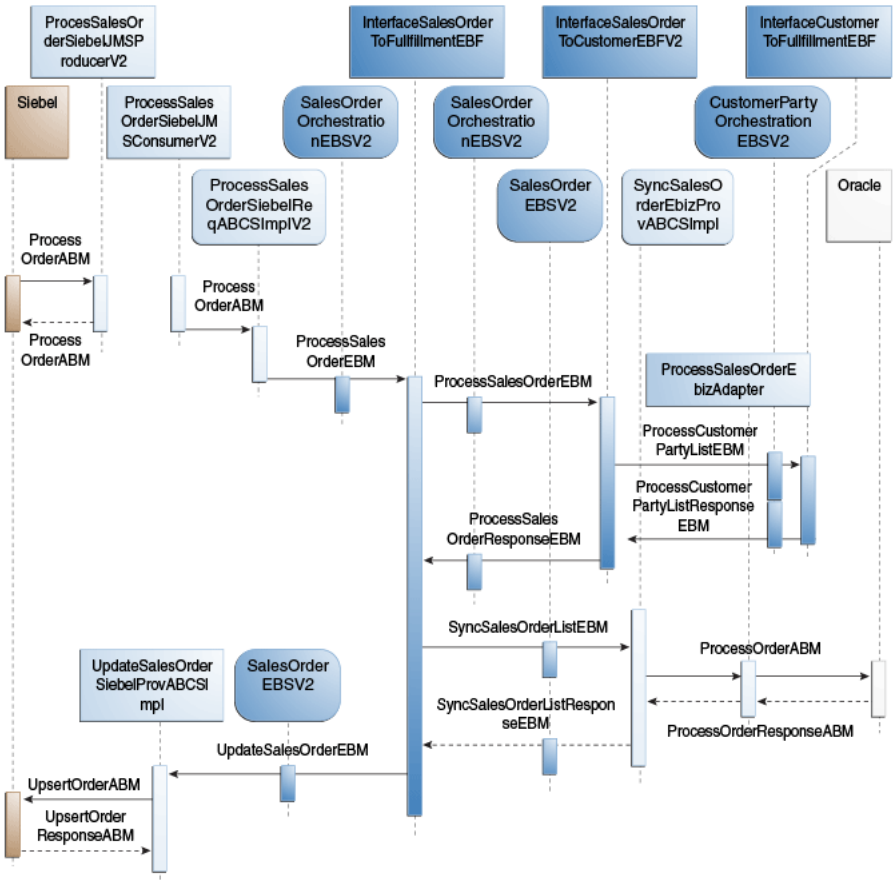
This integration flow uses the following interfaces:

- ProcessSalesOrderSiebelJMSProducerV2
- ProcessSalesOrderSiebelReqABCSImplV2
- SalesOrderOrchestrationEBSV2
- InterfaceSalesOrderToFulfillmentEBF
- InterfaceSalesOrderToCustomerEBF
- InterfaceCustomerToFulfillmentEBF
- SalesOrderEBSV2
- SyncSalesOrderEbizProvABCSImpl

- ProcessSalesOrderEbizAdapter
- UpdateSalesOrderSiebelProvABCSImpl

Figure 11–7 illustrates the Revise Sales Order (Siebel CRM-initiated) integration scenario

Figure 11–7 Revise Sales Order Flow (Siebel CRM - Initiated) Sequence Diagram



When you initiate the revise sales order process in Siebel CRM, the following events take place:

1. When an order is revised and submitted in Siebel CRM (version 8.0.0.x), workflow invokes the ProcessSalesOrderSiebelJMSProducerV2 service. If Siebel CRM 8.1.1.x is being used, then Siebel directly enqueues the message onto the AIA queue.
2. The invoked JMSProducer service enqueues the Siebel ABM to a Java Message Service (JMS) queue and replies the success or failure of the enqueue to the Siebel application.
3. The ProcessSalesOrderSiebelJMSConsumerV2 service dequeues the Siebel ABM from the JMS queue and invokes the ProcessSalesOrderSiebelReqABCSImplV2 service. For Siebel 8.1.1.x, the dequeue is done by ProcessSalesOrderSoapMsgSiebelJMSConsumer.
4. The invoked ProcessSalesOrderSiebelReqABCSImplV2 service transforms the Siebel ABM into the ProcessSalesOrderEBM.

   In doing so, the Account, Contact, and Address cross-reference tables are looked up or populated with the respective common IDs; the Order and OrderLine are

populated for the new order and for the revised order but not for the canceled order.

5. The ProcessSalesOrderSiebelReqABCSImplV2 then invokes the ProcessSalesOrder operation of the SalesOrderOrchestrationEBSV2 that is routed to the InterfaceSalesOrderToFulfillmentEBF enterprise business flow.
6. The InterfaceSalesOrderToFulfillmentEBF process in turn invokes the InterfaceSalesOrderToCustomerEBF process only for sales orders.

   A configuration parameter is available to prevent the customer sync service from invoking if a customer chooses. However, if it is decided based on a flag sent by Siebel whether the Address/Contact has changed, only then should the customer sync be invoked.
7. If customer sync is invoked, the InterfaceSalesOrderToCustomerEBF process transforms the SalesOrderEBM into the ProcessCustomerPartyListEBM containing a reduplicated list of the Account, Contact, and Address IDs that were referenced on the order.
8. The InterfaceSalesOrderToCustomerEBF then invokes the InterfaceCustomerToFulfillmentEBF that performs the necessary steps for synchronizing the customer accounts, contacts, and addresses to the fulfillment system (if the customer sync is invoked).
9. Upon receiving the response from the InterfaceCustomerToFulfillmentEBF service, the InterfaceSalesOrderToCustomerEBF sends a response EBM back to the InterfaceSalesOrderToFulfillmentEBF (if customer sync is invoked).
10. Upon receiving the response from the InterfaceSalesOrderToCustomerEBF service, the InterfaceSalesOrderToFulfillmentEBF performs a transformation to generate the SyncSalesOrderListEBM that is used to invoke the SyncSalesOrderList operation of the SalesOrderEBSV2.
11. The SalesOrderEBSV2 routes the SyncSalesOrderList invocation to the SyncSalesOrderEbizProvABCSImpl service.
12. The SyncSalesOrderEbizProvABCSImpl service transforms the SyncSalesOrderListEBM into the Oracle ProcessOrderABM.
13. The SyncSalesOrderEbizProvABCSImpl service then invokes the ProcessOrder operation of the ProcessSalesOrderEbizAdapter service.

    This service invokes the appropriate Oracle ProcessOrder PL/SQL application programming interface (API) that results in the update of the order in the Oracle EBS system.
14. Upon completion and response from the ProcessSalesOrderEbizAdapter, the SyncSalesOrderEbizProvABCSImpl generates the response EBM and replies to the SalesOrderResponseEBSV2 that in turn is routed back to the InterfaceSalesOrderToFulfillmentEBF.
15. The InterfaceSalesOrderToFulfillmentEBF then performs another transformation to generate the UpdateSalesOrderEBM that is used to invoke the UpdateSalesOrder operation of the SalesOrderEBSV2.
16. The SalesOrderEBSV2 routes the UpdateSalesOrder invocation to the UpdateSalesOrderSiebelProvABCSImpl service.
17. The UpdateSalesOrderSiebelProvABCSImpl transforms the UpdateSalesOrderEBM to the UpsertOrder Siebel ABM using the appropriate cross-reference tables to determine the Siebel IDs from Common IDs.

18. The UpdateSalesOrderSiebelProvABCSImpl then invokes the Siebel UpsertOrder web service to update the status of the order header.

11.5 Sales Order Cancellation

If any order is canceled in the order capture system (Siebel CRM), it must also be canceled in the fulfillment system (Oracle EBS). The order cancellation flow requests to cancel the entire order, canceling all open lines in an order. In this scenario, the order-level cancellation is initiated in Siebel CRM through an order revision, to be processed in Oracle EBS. If the fulfillment system (Oracle EBS) rejects the order cancellation for any reason, the order revision in Siebel CRM must be rolled back to prior version.

Additionally, a cancel reason for the order cancellation must also be provided.

11.5.1 Solution Assumptions and Constraints

The process integration for Order Management has the following assumptions:

1. The order can only be canceled if the order status is *Booked*.
2. Entire order cancellation is not possible if any of the order line status is *Shipped, Fulfilled, Closed, Cancelled, Cancel Pending*, or *Supply Eligible*.

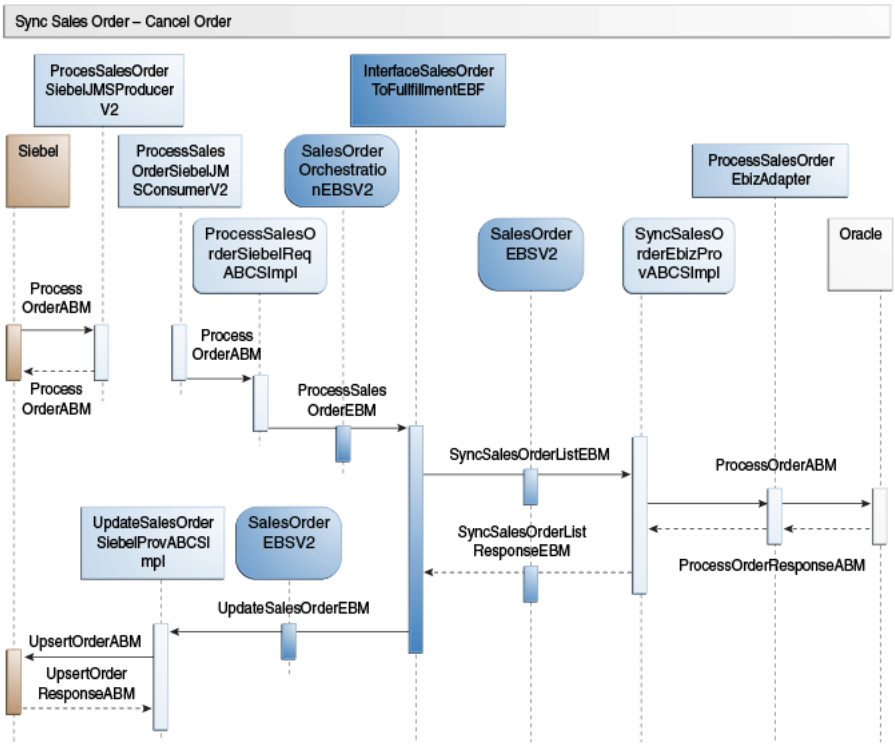
11.5.2 Cancel Sales Orders Integration Flow

This integration flow uses the following interfaces:

- ProcessSalesOrderSiebelJMSProducerV2
- ProcessSalesOrderSiebelReqABCSImplV2
- SalesOrderOrchestrationEBSV2
- InterfaceSalesOrderToFulfillmentEBF
- SalesOrderEBSV2
- SyncSalesOrderEbizProvABCSImpl
- ProcessSalesOrderEbizAdapter
- SalesOrderResponseEBSV2
- UpdateSalesOrderSiebelProvABCSImpl

Figure 11–8 illustrates the Cancel Sales Order integration scenario

Figure 11–8 Cancel Sales Order Flow Sequence Diagram



When you initiate the Cancel Sales Order process, the following events take place:

1. When an order is submitted for cancellation in Siebel CRM (version 8.0.0.x), workflow invokes the ProcessSalesOrderSiebelJMSProducerV2 service. If Siebel CRM 8.1.1.x is being used, then Siebel directly enqueues the message onto the AIA queue.
2. The invoked JMSProducer service enqueues the Siebel ABM (holding only header parameters) to a JMS queue and communicates the success or failure of the enqueue to the Siebel application.
3. The ProcessSalesOrderSoapMsgSiebelJMSConsumer service dequeues the Siebel ABM from the JMS queue and invokes the ProcessSalesOrderSiebelReqABCSImplV2 service. For Siebel 8.0.0.x, the dequeue is done by the ProcessSalesOrderSiebelJMSConsumerV2.
4. The invoked ProcessSalesOrderSiebelReqABCSImplV2 service transforms the Siebel ABM into the ProcessSalesOrderEBM.

   In doing so, the Order cross-reference table is looked up for the Siebel Row IDs and newly generated Common IDs.
5. The ProcessSalesOrderSiebelReqABCSImplV2 then invokes the ProcessSalesOrder operation of the SalesOrderOrchestrationEBSV2 that is routed to the InterfaceSalesOrderToFulfillmentEBF enterprise business flow.
6. InterfaceSalesOrderToFulfillmentEBF performs a transformation to generate the SyncSalesOrderListEBM that is used to invoke the SyncSalesOrderList operation of the SalesOrderEBSV2.
7. The SalesOrderEBSV2 routes the SyncSalesOrderList invocation to the SyncSalesOrderEbizProvABCSImpl.

8. The SyncSalesOrderEbizProvABCSImpl service transforms the SyncSalesOrderListEBM into the Oracle ProcessOrderABM.
9. The SyncSalesOrderEbizProvABCSImpl service then invokes the ProcessOrder operation of the ProcessSalesOrderEbizAdapter service.

   This service invokes the appropriate Oracle ProcessOrder PL/SQL API that results in the cancellation of the order in the Oracle EBS system.
10. Upon completion and response from the ProcessSalesOrderEbizAdapter, the SyncSalesOrderEbizProvABCSImpl generates the response EBM, during which the Oracle IDs are looked up in the cross-reference, and replies to the SalesOrderResponseEBSV2, which in turn is routed back to the InterfaceSalesOrderToFulfillmentEBF.
11. The InterfaceSalesOrderToFulfillmentEBF then performs another transformation to generate the UpdateSalesOrderEBM, which is used to invoke the UpdateSalesOrder operation of the SalesOrderEBSV2.
12. The SalesOrderEBSV2 routes the UpdateSalesOrder invocation to the UpdateSalesOrderSiebelProvABCSImpl service.
13. The UpdateSalesOrderSiebelProvABCSImpl transforms the UpdateSalesOrderEBM to the UpsertOrder Siebel ABM using the appropriate cross-reference tables to determine the Siebel IDs from Common IDs.
14. The UpdateSalesOrderSiebelProvABCSImpl then invokes the Siebel UpsertOrder web service to update the status of the order header.

11.6 Siebel CRM Interfaces

These are the Siebel CRM web services for the Order Management integration flow:

Inbound Siebel CRM Web Services

- Service Name: SWIOrderUpsert
  - Operation Name: SBLOrderUpsert
  - Request Schema: SWIOrderIO.xsd
  - Response Schema: SWIOrderIO.xsd
- Service Name: OrderUpsertService

  The UpdateSalesOrderSiebelProvABCSImpl invokes this service during:
  - The CreateOrder integration flow after the order has been synced to back office. The order header's integration status is updated to Created In Back Office or Error In Back Office.
  - The Update Order integration flow after the order line update is done in the back office. UpdateSalesOrderSiebelProvABCSImpl also invokes OrderUpsertService to update shipping details of the order line.
- Service Name: QuoteUpsertService

  The UpdateSalesOrderSiebelProvABCSImpl invokes this service during the CreateQuote integration flow after syncing the quote to the back office. The quote header's integration status is updated to Created In Back Office or Error In Back Office.

Outbound Siebel CRM Web Services

- Sales Order Submitted - Siebel CRM invokes ProcessSalesOrderSiebelJMSProducerV2 with the ListOfSWIOrderIO ABM
- Quote Submitted - Siebel CRM invokes the ProcessQuoteSiebelJMSProducer with the ListOfSWIQuoteIO ABM
- New Sales Order, Revised Order, or Cancel Order Submitted - Siebel CRM invokes the ProcessSalesOrderSiebelJMSProducerV2 service.

For more information about Siebel web services, see *CRM Web Services Reference*.

11.7 Oracle EBS Interfaces

These are the Oracle EBS web services for the Order Management integration flow:

Inbound to Oracle EBS Web Services

- OE_INBOUND_INT.PROCESS_ORDER_25 (Process Sales Order Service)
- OE_OUTBOUND_INT.SYNC_ORDER (Get Sales Order Service)
- GetSalesOrderLineShippingDetailsService

  The UpdateSalesOrderEbizReqABCSImpl service invokes this service during the Update Order integration flow to retrieve the order line shipping details before sending the updates to Siebel.
- GetSalesOrderService

  The UpdateSalesOrderEbizReqABCSImpl service invokes this service during the Update Order integration flow to retrieve the order header or line update details before sending the updates to Siebel.

Outbound from Oracle EBS Event Interfaces

- oracle.apps.ont.genesis.outbound.update

For more information about Oracle E-Business Suite web services and documentation prior to Release 12.1.3, see the library on Oracle Technology Network: http://www.oracle.com/technetwork/documentation/applications-167706.html?. For Oracle E-Business Suite documentation for R12.1.3 and beyond, see this library: http://download.oracle.com/docs/cd/E18727_01/index.htm?

11.8 Core Oracle AIA Components

The Order Management process integration uses the following delivered EBOs and enterprise business messages (EBMs):

- SalesOrderEBO
- ProcessSalesOrderEBM
- ProcessSalesOrderResponseEBM
- CreateSalesOrderEBM
- CreateSalesOrderResponseEBM
- UpdateSalesOrderEBM
- UpdateSalesOrderResponseEBM
- ProcessCustomerPartyListEBM

- ProcessCustomerPartyListResponseEBM

The industry enterprise business object (EBO) and enterprise business message XML schema (EBM XSD) files are located here: $AIA_HOME/AIAMetaData/AIAComponents/EnterpriseObjectLibrary/Core/EBO/

The industry enterprise business service (EBS) WSDL files are located here: $AIA_HOME/AIAMetaData/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/

For detailed documentation of individual EBOs and EBMs, click the AIA Reference Doc link on EBO and EBM detail pages in Oracle Enterprise Repository.

For more information about using the Oracle Enterprise Repository and configuring it to provide the AIA Reference Doc link, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring and Using Oracle Enterprise Repository as the Oracle AIA SOA Repository."

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

For more information, see *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Extensibility."

11.9 Integration Services

The integration provides these services:

- SalesOrderEBSV2
- SalesOrderResponseEBSV2
- SalesOrderOrchestrationEBSV2
- SalesOrderOrchestrationResponseEBSV2
- CustomerPartyOrchestrationEBSV2
- CustomerPartyOrchestrationResponseEBSV2
- InterfaceSalesOrderToFulfillmentEBF
- InterfaceSalesOrderToCustomerEBFV2
- ProcessSalesOrderSiebelJMSProducerV2
- ProcessQuoteSiebelJMSProducer
- ProcessSalesOrderSiebelJMSConsumerV2
- ProcessSalesOrderSoapMsgSiebelJMSConsumer
- ProcessQuoteSiebelJMSConsumer
- ProcessQuoteSoapMsgSiebelJMSConsumer
- ProcessSalesOrderSiebelReqABCSImplV2
- ProcessQuoteSiebelReqABCSImpl
- CreateSalesOrderEbizProvABCSImpl
- SyncSalesOrderEbizProvABCSImpl
- UpdateSalesOrderEbizEventConsumer
- UpdateSalesOrderEbizReqABCSImpl

- UpdateSalesOrderSiebelProvABCSImpl
- ProcessSalesOrderEbizAdapter
- GetSalesOrderEbizAdapter
- GetSalesOrderLineShippingDetailsEbizAdapter

11.9.1 SalesOrderEBSV2

The SalesOrderEBSV2 is an EBS that provides basic request and response operations that can be performed against the SalesOrderEBO. This service is invoked as part of the create order, update order, and cancel order integration flows.

SalesOrderEBSV2 Operations

- CreateSalesOrder
- UpdateSalesOrder
- SyncSalesOrderList

The SalesOrderEBSV2 is implemented as a Mediator routing service.

For more information about this EBS, see the *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Designing and Developing Enterprise Business Services" and *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Enterprise Business Services."

11.9.2 SalesOrderResponseEBSV2

The SalesOrderResponseEBSV2 service is an EBS that provides the basic response operations that can be performed against the SalesOrderEBO. This service is invoked as part of the create order, update order, and cancel order integration flows.

SalesOrderResponseEBSV2 Operations

- CreateSalesOrderResponse
- UpdateSalesOrderResponse
- SyncSalesOrderListResponse

The SalesOrderResponseEBSV2 service is implemented as a Mediator routing service.

For more information about this EBS, see the *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Designing and Developing Enterprise Business Services" and *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Enterprise Business Services."

11.9.3 SalesOrderOrchestrationEBSV2

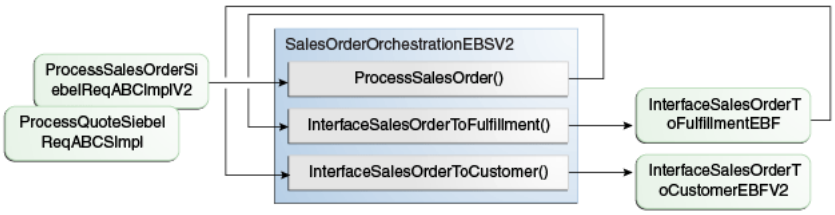
The SalesOrderOrchestrationEBSV2 and SalesOrderOrchestrationResponseEBSV2 are enterprise business services providing request and response operation routing to process a sales order through a business process flow. These services are invoked as part of the create order, change order, and cancel order integration flows

---

Note: Invoking InterfaceSalesOrderToCustomer for cancel order is not required, because the order is not revised.

Figure 11–9 illustrates the request and response operation routing to process a sales order.

Figure 11–9 SalesOrderOrchestrationEBSV2 Routing Diagram



The SalesOrderOrchestrationResponseEBSV2 service exposes the asynchronous response operations for each of the request operations.

The SalesOrderOrchestrationEBSV2 service has four asynchronous one-way operations. The SalesOrderOrchestrationResponseEBSV2 has four asynchronous response operations.

SalesOrderOrchestrationEBSV2 Operations

- ProcessSalesOrder
  - This operation is routed to the InterfaceSalesOrderToFulfillment operation of the same EBS.
  - If customers want to insert a custom orchestration process into the flow, they must define a routing rule that routes this operation to the custom orchestration process rather than the InterfaceSalesOrderToFulfillment.
- InterfaceSalesOrderToFulfillment

  This operation is routed to the InterfaceSalesOrderToFulfillmentEBF.
- InterfaceSalesOrderToCustomer

  This operation is routed to the InterfaceSalesOrderToCustomerEBFV2.

SalesOrderOrchestrationResponseEBSV2 Operations

- ProcessSalesOrderResponse

  This operation is intended to be routed to the caller of the ProcessSalesOrder operation as indicated in the EBM header. However, no routing targets are provided because the ProcessSalesOrderSiebelReqABCSImpl does not expect to receive a response.
- InterfaceSalesOrderToFulfillmentResponse

  This operation is routed to the ProcessSalesOrderResponse operation as indicated in the EBM header.
- InterfaceSalesOrderToCustomerResponse

  This operation is routed to the caller of the InterfaceSalesOrderToCustomer operation (InterfaceSalesOrderToFulfillmentEBF) as indicated in the EBM header.

For more information about this EBS, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Designing and Developing Enterprise Business Services" and *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Enterprise Business Services."

11.9.4 CustomerPartyOrchestrationEBSV2

The CustomerPartyOrchestrationEBSV2 service is an EBS that provides a request operation that can be performed against the order to sync the customer information by invoking InterfaceCustomerToFulfillmentEBF. This service is invoked as part of the Create Order.

CustomerPartyOrchestrationEBSV2 Operations

InterfaceCustomerToFulfillment

For more information about this EBS, see the *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Designing and Developing Enterprise Business Services" and *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Enterprise Business Services."

11.9.5 CustomerPartyOrchestrationResponseEBSV2

The CustomerPartyOrchestrationResponseEBSV2 service is an EBS that provides a response operation that is used to return an InterfaceCustomerToFulfillmentEBF response to InterfaceSalesOrderToFulfillmentEBF. This service is invoked as part of the Create Order.

CustomerPartyOrchestrationResponseEBSV2 Operations

InterfaceCustomerToFulfillmentResponse

For more information about this EBS, see the *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Designing and Developing Enterprise Business Services" and *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Enterprise Business Services."

11.9.6 InterfaceSalesOrderToFulfillmentEBF

The InterfaceSalesOrderToFulfillmentEBF service is an enterprise business flow that interfaces a sales order to a back-office fulfillment system. This service is invoked as part of the create order, change order, and cancel order integration flows.

This process performs three high-level actions:

1. Interfaces customer accounts from the order to the fulfillment system.

   The InterfaceSalesOrderToCustomerEBFV2 is invoked for this. This step can be configured so that it can be suppressed.

2. Creates, changes, or cancels the order in the fulfillment system using the SalesOrderEBSV2 Create or SyncSalesOrderList operations.

3. Updates the order status in the source order capture system.

   This step can be configured so that it can be suppressed.

The InterfaceSalesOrderToFulfillmentEBF is an asynchronous Business Process Execution Language (BPEL) process. Upon completion, it invokes the InterfaceSalesOrderToFulfillmentResponse operation of the SalesOrderOrchestrationResponseEBSV2.

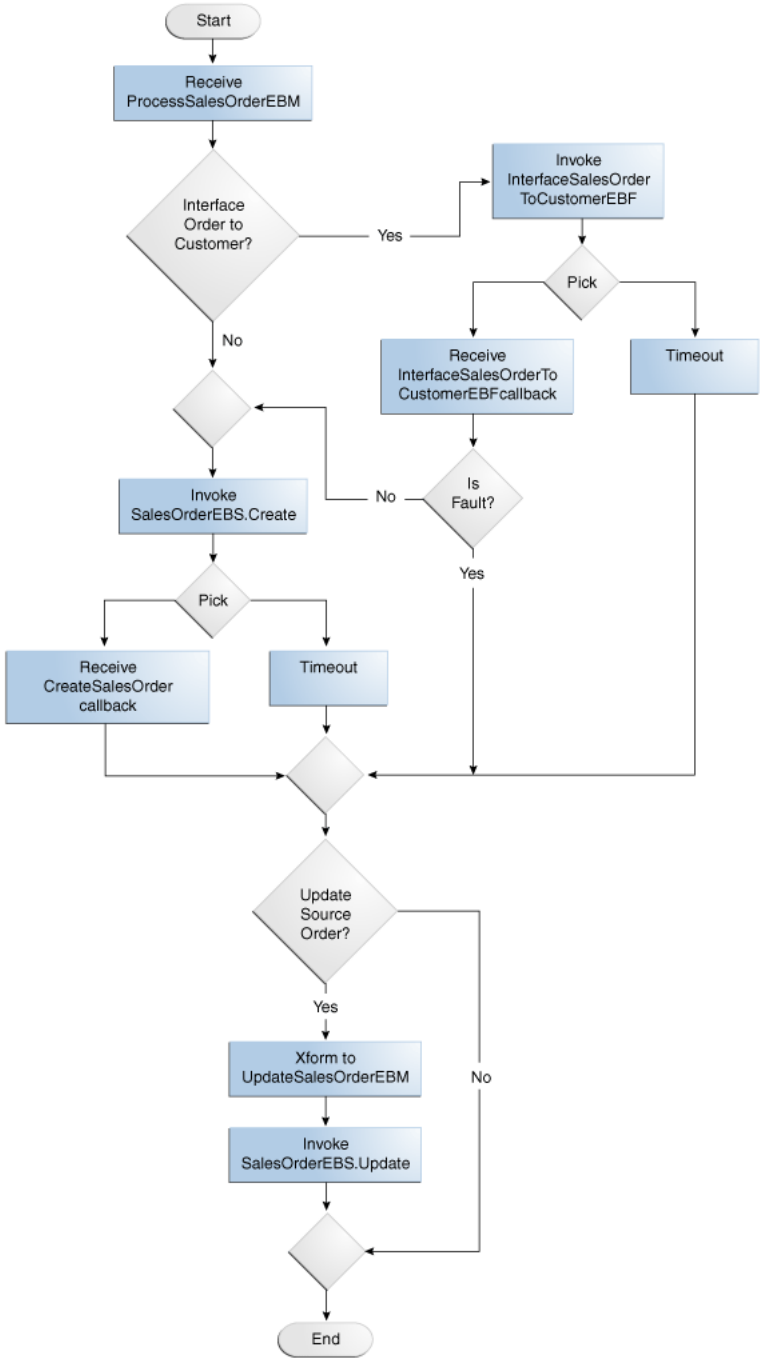
This enterprise business flow (EBF) has five inbound operations. The first initiates the process, and the remaining operations receive the asynchronous callbacks from the other service operations that this process invokes. These are the inbound operations:

- InterfaceSalesOrderToFulfillment
- InterfaceSalesOrderToCustomerResponse
- CreateSalesOrderResponse
- UpdateSalesOrderResponse
- SyncSalesOrderListResponse

Figure 11–10 illustrates the InterfaceSalesOrderToFulfillmentEBF activity flow:

InterfaceSalesOrderToFulfillmentEBF activity flow diagram

Figure 11–10 InterfaceSalesOrderToFulfillmentEBF Activity Flow Diagram



These are the transformations:

- ProcessSalesOrderEBM to CreateSalesOrderEBM
- ProcessSalesOrderEBM + CreateSalesOrderResponseEBM to UpdateSalesOrderEBM
- ProcessSalesOrderEBM to SyncSalesOrderListEBM
- ProcessSalesOrderEBM + SyncSalesOrderListResponseEBM to UpdateSalesOrderEBM

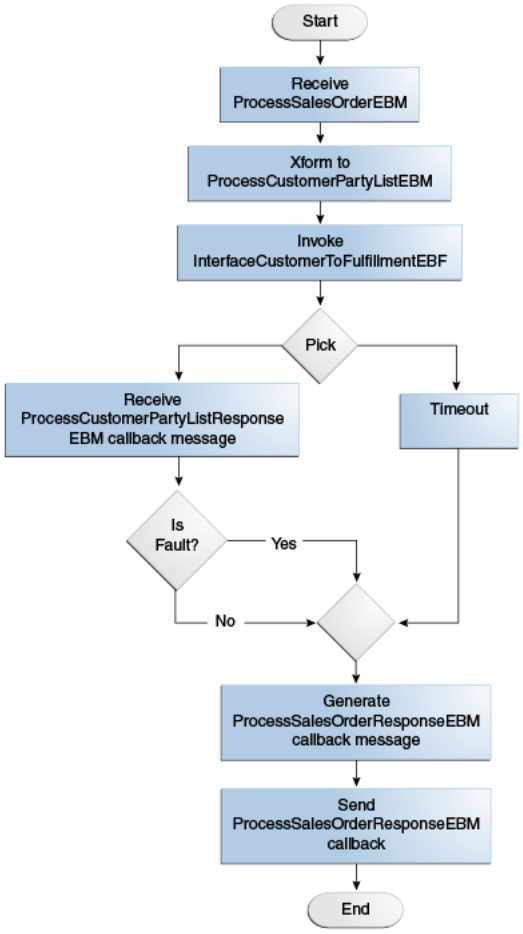
11.9.7 InterfaceSalesOrderToCustomerEBFV2

The InterfaceSalesOrderToCustomerEBFV2 service is an enterprise business flow that extracts the list of distinct customer accounts, addresses, and contacts from the order and invokes the InterfaceCustomerToFulfillmentEBF service. This service is invoked as part of the Create Order or Sync Revised Order integration flow.

The InterfaceSalesOrderToCustomerEBFV2 enterprise business flow is implemented as an asynchronous request+callback BPEL process.

Figure 11–11 illustrates the InterfaceSalesOrderToCustomerEBFV2 activity flow:

Figure 11–11 InterfaceSalesOrderToCustomerEBFV2 Activity Flow Diagram



These are the transformations:

- ProcessSalesOrderEBM to ProcessCustomerPartyListEBM: this transformation must pass a list of accounts referenced in the order (for example, the order header-level and line-level accounts) and the bill-to and ship-to addresses referenced for each account.

  This list must not contain duplicates.

- ProcessSalesOrderEBM + ProcessCustomerPartyListResponseEBM to ProcessSalesOrderResponseEBM: this transformation passes through success or error messages.

11.9.8 ProcessSalesOrderSiebelJMSProducerV2

The ProcessSalesOrderSiebelJMSProducerV2 service guarantees delivery of the ProcessSalesOrder Siebel ABM to the JMS queue. This service is invoked synchronously from the Siebel application workflow. The response indicates whether the message was successfully enqueued. This service is invoked as part of the Create Order integration flow when the version of the Siebel CRM application is 8.0.0.x.

The assumption is that the Siebel order was validated from within the Siebel workflow.

The ProcessSalesOrderSiebelJMSProducerV2 service has a single synchronous request+reply operation: ProcessSalesOrder.

11.9.9 ProcessSalesOrderSiebelJMSConsumerV2

The ProcessSalesOrderSiebelJMSConsumerV2 service dequeues the ProcessSalesOrder Siebel ABM from the JMS queue and asynchronously invokes the ProcessSalesOrderSiebelReqABCSImpl service. This service is invoked as part of the Create Order integration flow when the version of the Siebel CRM application is 8.0.0.x.

The ProcessSalesOrderSiebelJMSConsumerV2 service has an inbound JMS adapter front end, which subscribes to the JMS queue.

The ProcessSalesOrderSiebelJMSConsumerV2 service is implemented as an inbound JMS adapter service in Mediator.

11.9.10 ProcessSalesOrderSoapMsgSiebelJMSConsumer

The ProcessSalesOrderSoapMsgSiebelJMSConsumer service dequeues the ProcessSalesOrder Siebel ABM from the JMS queue and asynchronously invokes the ProcessSalesOrderSiebelReqABCSImpl service. This service is invoked as part of the Create Order integration flow when the version of the Siebel CRM application is 8.1.1.x.

The ProcessSalesOrderSoapMsgSiebelJMSConsumer service has an inbound JMS adapter front-end, which subscribes to the JMS queue.

The ProcessSalesOrderSoapMsgSiebelJMSConsumer service is implemented as an inbound JMS adapter service in Mediator.

11.9.11 ProcessSalesOrderSiebelReqABCSImplV2

The ProcessSalesOrderSiebelReqABCSImplV2 service transforms the ProcessSalesOrder Siebel ABM into the canonical ProcessSalesOrderEBM and asynchronously invokes the ProcessSalesOrder operation of the SalesOrderOrchestrationEBSV2 to initiate the InterfaceSalesOrderToFulfillmentEBFV2. This service is invoked as part of the create order, change order, and cancel order integration flows.

- In the create order integration flow: As part of the transformation to the ProcessSalesOrderEBM, common IDs for the order, order lines, referenced accounts, referenced ship to and bill to addresses, and contacts are generated and loaded into the cross-reference.
- In the revise or change order integration flow: As part of the transformation to the ProcessSalesOrderEBM, common IDs for the order and order lines are regenerated and loaded into the cross-reference to point to new Siebel Row IDs, and referenced

accounts, ship to, bill to addresses, and contacts are entered or looked up from the cross-reference.

- In the cancel order integration flow: As part of the transformation to the ProcessSalesOrderEBM, common IDs for an order are looked up from the cross-reference.

The ProcessSalesOrderSiebelReqABCSImplV2 service has a single asynchronous request-only operation: ProcessSalesOrder. It accepts the Siebel Order ABM.

The one transformation is ProcessSalesOrderABM to ProcessSalesOrderEBM.

The ProcessSalesOrderSiebelReqABCSImplV2 application business connector service is implemented as an asynchronous request-only BPEL process.

11.9.12 CreateSalesOrderEbizProvABCSImpl

The CreateSalesOrderEbizProvABCSImpl service provides the Oracle EBS implementation for the CreateSalesOrder operation of the SalesOrderEBSV2. This service is invoked as part of the create order integration flow by the CreateSalesOrder operation of the SalesOrderEBSV2.

This service invokes the Process Sales Order PL/SQL API in Oracle EBS using the ProcessSalesOrderEbizAdapter service registered in Mediator.

The CreateSalesOrderEbizProvABCSImpl service is the Oracle EBS provider for the CreateSalesOrder operation of the SalesOrderEBSV2. When complete, this service invokes the CreateSalesOrderResponse operation of the SalesOrderResponseEBSV2.

These are the transformations:

- CreateSalesOrderEBM to ProcessSalesOrder ABM.
- ProcessSalesOrderResponse ABM to CreateSalesOrderResponseEBM.

The CreateSalesOrderEbizProvABCSImpl application business connector service is implemented as an asynchronous BPEL process.

11.9.13 SyncSalesOrderEbizProvABCSImpl

The SyncSalesOrderEbizProvABCSImpl service provides the Oracle EBS implementation for the SyncSalesOrderList operation of the SalesOrderEBSV2. This service is invoked as part of the create order, change order, and cancel order integration flows through the SyncSalesOrderList operation of the SalesOrderEBSV2.

This service in turn invokes the Process Sales Order PL/SQL API in Oracle through the ProcessSalesOrderEbizAdapter service registered in Mediator.

These are the transformations:

- SyncSalesOrderListEBM to ProcessSalesOrder ABM
- ProcessSalesOrderResponse ABM to SyncSalesOrderListResponseEBM

The SyncSalesOrderEbizProvABCSImpl application business connector service (ABCS) is implemented as an asynchronous BPEL process.

11.9.14 UpdateSalesOrderEbizEventConsumer

The UpdateSalesOrderEbizEventConsumer service subscribes to the oracle.apps.ont.genesis.outbound.update business event in Oracle EBS. After the event is picked up from Oracle AQ, it is passed to the UpdateSalesOrderEbizReqABCSImpl service.

This set of fields uniquely identifies the event:

- HEADER_ID
- LINE_ID
- HDR_REQ_ID
- LIN_REQ_ID
- CHANGE_TYPE
- HOLD_SOURCE_ID
- ORDER_HOLD_ID

This service is an inbound AQ adapter service and does not have a public interface. The service is initiated by Mediator when the subscription event occurs.

This service is implemented as an inbound-to-SOA Oracle Apps Event adapter service in Mediator.

11.9.15 UpdateSalesOrderEbizReqABCSImpl

The UpdateSalesOrderEbizReqABCSImpl service fetches the Oracle Order ABM from GetSalesOrderEbizAdpater based on event payload. For any Order Line whose status is Shipped, the OrderLineShippingDetailsABM is fetched from the GetSalesOrderLineShippingDetailsEbizAdapter. The combined Oracle order ABMs are transformed into UpdateSalesOrderEBM and then the UpdateSalesOrderEbizReqABCSImpl invokes the UpdateSalesOrder operation of the SalesOrderEBSV2.

This service has one asynchronous request operation that accepts the Oracle event ABM.

These are the transformations:

- WF_EVENT_T_msg à args_in_msg (GetSalesOrderEbizAdpater)
- Args_out_msg (GetSalesOrderEbizAdpater) à args_in_msg (GetSalesOrderLineShippingDetailsEbizAdapter)
- Args_out_msg (GetSalesOrderEbizAdpater) + args_out_msg (GetSalesOrderLineShippingDetailsEbizAdapter) à UpdateSalesOrderEBM

The UpdateSalesOrderEbizReqABCSImpl service is implemented as an asynchronous fire-and-forget BPEL process.

11.9.16 UpdateSalesOrderSiebelProvABCSImpl

The UpdateSalesOrderSiebelProvABCSImpl service is the Siebel provider ABCS for the UpdateSalesOrder operation of the SalesOrderEBS. This service invokes the Siebel Upsert Order web service or the Upsert Quote web service, depending on the nature of the update and type of order (for example, sales order or RMA or quote).

This service implements the UpdateSalesOrder operation defined in the SalesOrderEBSV2 service. This operation is an asynchronous request operation that accepts the UpdateSalesOrderEBM. After completion, this service invokes the UpdateSalesOrderResponse operation of the SalesOrderResponseEBSV2.

These are the transformations:

- UpdateSalesOrderEBM to UpsertOrderABM
- UpdateSalesOrderEBM to UpsertQuoteABM

- UpsertOrder Response ABM to UpdateSalesOrderResponseEBM
- UpsertQuote Response ABM to UpdateSalesOrderResponseEBM

11.9.17 ProcessSalesOrderEbizAdapter

The ProcessSalesOrderEbizAdapter service is an Oracle EBS Adapter service registered in Mediator. This adapter service exposes the Oe_Inbound_Int.Process_Order_25 PL/SQL. This service is the interface through which an order is created in Oracle EBS and is invoked by the CreateSalesOrderEbizProvABCSImpl / SyncSalesOrderEbizProvABCSImpl as part of the create order, change order, and cancel order integration flows.

The ProcessSalesOrderEbizAdapter service exposes the Process Order operation of the PL/SQL API. This operation is a synchronous request+reply operation. By registering this adapter service in Mediator, Mediator exposes a Simple Object Access Protocol (SOAP) binding that is used in this integration to invoke the service from the CreateSalesOrderEbizProvABCSImpl and SyncSalesOrderEbizProvABCSImpl.

The service is implemented as an EBS adapter service in Mediator.

11.9.18 GetSalesOrderEbizAdpater

The GetSalesOrderEbizAdpater service is an Oracle EBS Adapter service registered in Mediator. This adapter service exposes the Oe_Outbound_Int.Sync_Order25 PL/SQL API delivered as part of EBS12.1.1. This service is invoked as part of the Update Order flow initiated when an order update event is launched in Oracle EBS.

This set of fields uniquely identifies the event:

- HEADER_ID
- LINE_ID
- HDR_REQ_ID
- LIN_REQ_ID
- CHANGE_TYPE
- HOLD_SOURCE_ID
- ORDER_HOLD_ID

11.9.19 GetSalesOrderLineShippingDetailsEbizAdapter

The GetSalesOrderLineShippingDetailsEbizAdapter service is an Oracle EBS Adapter service registered in Mediator. This adapter service exposes the WSH_ INTEGRATION.Get_Delivery_Detail_attributes PL/SQL API delivered as part of EBS 11.5.10 and 12.1.

This service is invoked as part of the update order flow initiated when an order line status becomes shipped. This operation is a synchronous request and reply operation. Because this adapter service is registered in Mediator, Mediator exposes a SOAP binding that is used in this integration to invoke the service from the UpdateSalesOrderEbizReqABCSImpl.

12

Process Integration for Asset Management

This chapter provides an overview of the process integration for Asset Management and discusses how to create and update assets. Also discussed are Siebel Customer Relationship Management (Siebel CRM) and Oracle E-Business Suite (Oracle EBS) interfaces, core Oracle Application Integration Architecture (Oracle AIA) components, and integration services.

This chapter includes the following sections:

12.1 Process Integration for Asset Management

The process integration for Asset Management is a one-way synchronization of new or updated customer-owned item instances from Oracle EBS into a Siebel CRM asset. The asset integration supports the following flows:

- **Create assets:** This flow enables the synchronization of new customer-owned item instances from Oracle EBS to Siebel CRM Asset.
- **Update assets:** This flow enables the synchronization of updates to item instances from Oracle EBS to Siebel CRM Asset.
- **Bulk load of assets:** This flow enables the extraction, transformation, and loading (ETL) of initial Item instance data from Oracle EBS to Siebel CRM Asset. This feature uses Oracle Data Integrator (ODI) to extract relevant data from Oracle EBS and map it to Siebel Enterprise Integration Manager (EIM) interface tables. This process also enables cross-referencing between Oracle EBS and Siebel CRM.

For more information about initial data loads, see Chapter 2, "Loading Initial Data."

12.1.1 Prerequisites

Find prerequisites in Section 13.11, "Asset Management: Prerequisites and Data Requirements."

12.1.2 Solution Assumptions and Constraints

These are the assumptions and constraints for the asset integration:

1. After assets are synchronized to Siebel CRM, they cannot be changed in Siebel CRM.
2. In Oracle EBS, item instances are created and updated for items for which the Track in Installed Base attribute is set to *True*.
3. Only customer-owned (Organization Party) item instances (assets) are synchronized from Oracle EBS to Siebel CRM.
4. In Siebel CRM, the asset is created upon synchronization from Oracle EBS, and not by any workflow process in Siebel CRM when the order status is completed.
5. Because asset synchronization is one-way from Oracle EBS to Siebel CRM, assets should not be updated in Siebel CRM.
6. When products are synchronized from Oracle EBS to Siebel CRM, they are cross-referenced using this concatenation of keys from Oracle EBS:

   ```
   inventory item id::organization id::operating unit id
   TYPE = SHELL
   ```

   When orders are fulfilled and item instances are created in Oracle EBS, the master organization Id and the shipping organization Ids are stored along with the item instance details in Oracle EBS.

   When products are synchronized it should be ensured that they are also synchronized in the master organization and any other organizations, (such as the item validation organization), which Order Management requires. Then asset integration fetches the product from the cross-reference using the following:

   ```
   inventory item id::master org id::operating unit id
   ```

Figure 12–1 illustrates where the asset integration flow occurs in the Order to Cash pre-built integration.

Figure 12–1 Asset Management Integration Flow in Order to Cash: Siebel CRM - EBS

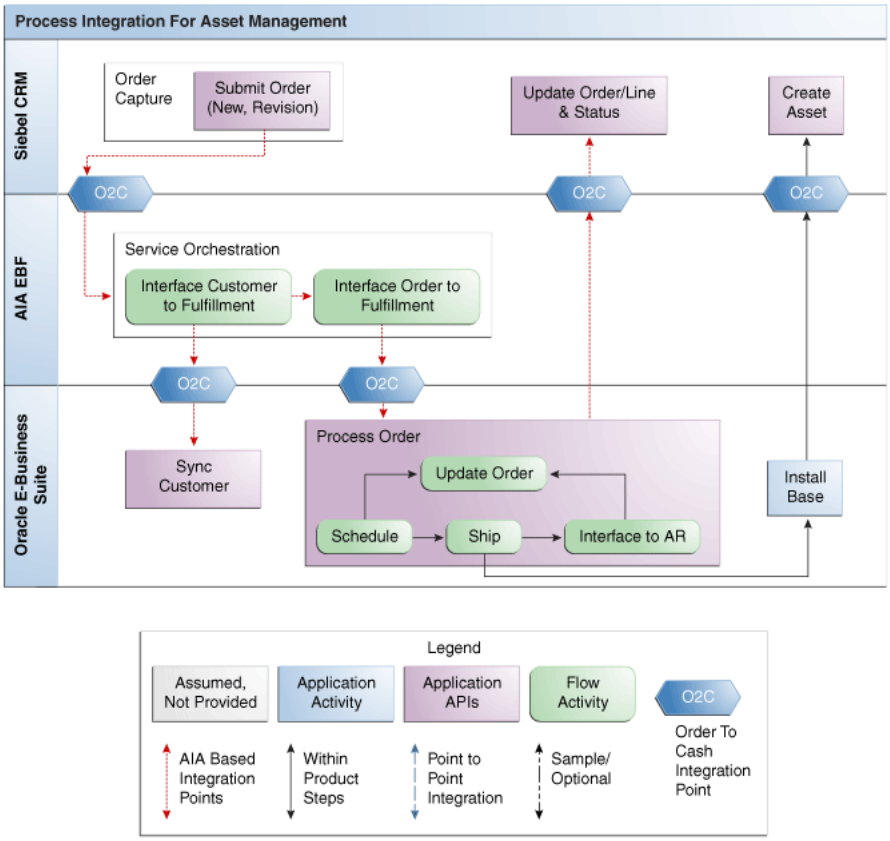
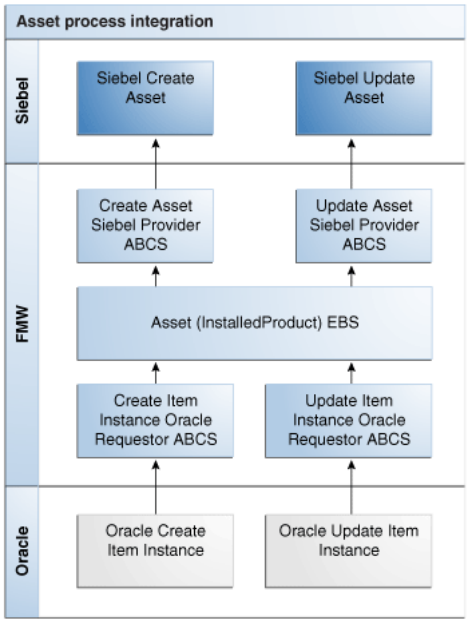


Figure 12–2 illustrates the Create Asset and Update Asset integration flows.

Figure 12–2 Create Asset and Update Asset Integration Flow



12.2 Creating Assets

The Create Assets integration flow synchronizes new item instances from Oracle EBS to Siebel CRM Assets in real time and it enables a Customer Service Representative (CSR) to share the asset information with customers.

The item instances are created in the following ways in Oracle EBS:

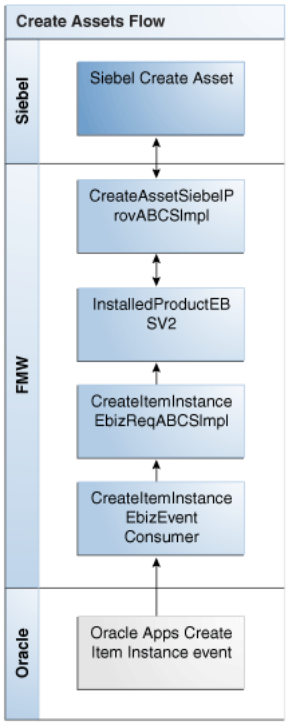
- After sales orders are picked, released, and shipped, an install base item instance is created in Oracle EBS (using auto-creation).
- An item instance can be manually created for a customer (for both serial and nonserial items).

The major attributes of the Oracle EBS Item Instance that are synchronized to Siebel CRM Asset are:

- Item (Product)
- Asset Number and Serial Number
- Owner Account
- Order Id
- Status and Install Date
- Quantity

Figure 12–3 illustrates the Create Assets integration flow.

Figure 12–3 Create Assets Integration Flow



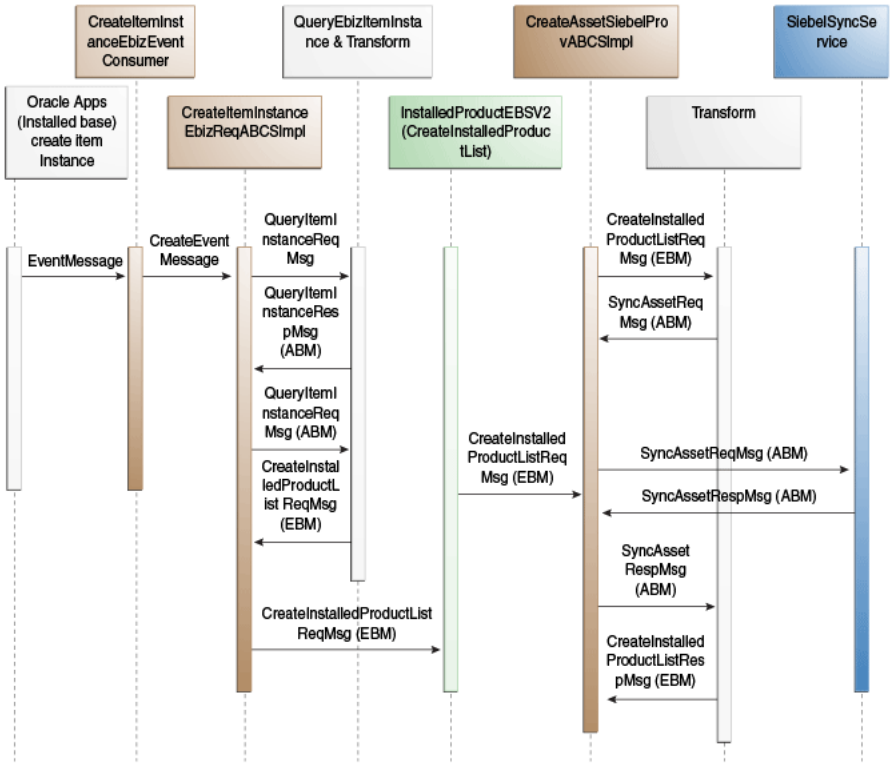
12.2.1 Create Assets Flow

This integration flow uses the following interfaces:

- CreateItemInstanceEbizEventConsumer
- QueryItemInstanceEbizAdapter
- CreateItemInstanceEbizReqABCSImpl
- InstalledProductEBSV2
- CreateAssetSiebelProvABCSImpl

Figure 12–4 illustrates the Create Assets integration scenario.

Figure 12–4 Create Assets Flow Sequence Diagram



When you initiate the create assets process, the following events occur:

1. The create event is raised when a new item instance is created in Oracle EBS either through the user interface or through the Order fulfillment process.

   The event is raised for simple and complex asset creations. The CreateItemInstanceEbizEventConsumer dequeues the message using the EBS adapter listening on the create item instance event. The event payload is then routed to the CreateItemInstanceEbizReqABCSImpl service.

2. Using the Oracle EBS adapter, the CreateItemInstanceEbizReqABCSImpl service queries CSI_ASSET_INTEGRATION_V passing in the instance ID and gets the entire item instance ABM message as QueryItemInstanceRespMsg.

3. The CreateItemInstanceEbizReqABCSImpl then transforms the QueryItemInstanceRespMsg into the CreateInstalledProductEBM, and then it invokes the InstalledProductEBSV2 with the operation CreateInstalledProductList.

   In the case of a complex asset, as part of the create transformation, the parent ID is ignored even if it is available in the message. It is mapped, and the relationship with the parent item instance is built only as part of the update process.

4. The InstalledProductEBSV2 routes the CreateInstalledProductEBM to the Siebel provider application business connector service (ABCS) implementation CreateAssetSiebelProvABCSImpl.
5. The CreateAssetSiebelProvABCSImpl transforms the CreateInstalledProductEBM into the Siebel Asset application business message (ABM) CRMIntegSEBLHORAssetInterface as CreateAssetReqMsg.

   It then calls the Siebel Asset web service CRMIntegSEBLHORAssetInterface to create the asset in Siebel and returns a response message CreateAssetRespMsg. The CreateAssetSiebelProvABCSImpl then transforms the Siebel response message to the enterprise business message (EBM) CreateInstalledProductRespMsg (the Siebel Asset ID links to the common ID in the cross-reference) and then the message is sent back to the InstalledProductEBSV2.

12.3 Updating Assets

The Update Assets integration flow is initiated when item instances are updated in Oracle EBS. Updates include status updates, updates due to return material authorization (RMA) orders, and relationship updates to build the hierarchy to represent the Bill of Material (BoM).

This one-way synchronization is from Oracle EBS to Siebel CRM. As item instances are updated in Oracle EBS, a business event is triggered that allows the synchronization of the latest asset status from Oracle EBS to Siebel CRM.

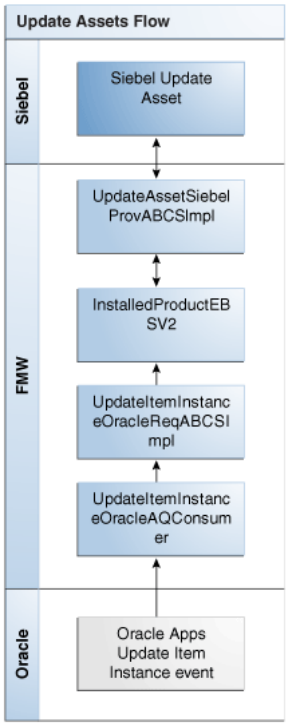
Item Instances are updated when:

- A shipped confirmation of an item instance (auto-update) is received.
- Merchandise is returned (RMA transaction). The install base item status is updated as returned, and the quantity is also updated.
- Manual updates of item instances occur (the update of attributes versus the retiring of assets).

Note: Asset updates are not allowed from Siebel CRM to Oracle EBS.

Figure 12–5 illustrates the Update Assets integration flow.

Figure 12–5 Update Assets Integration Flow



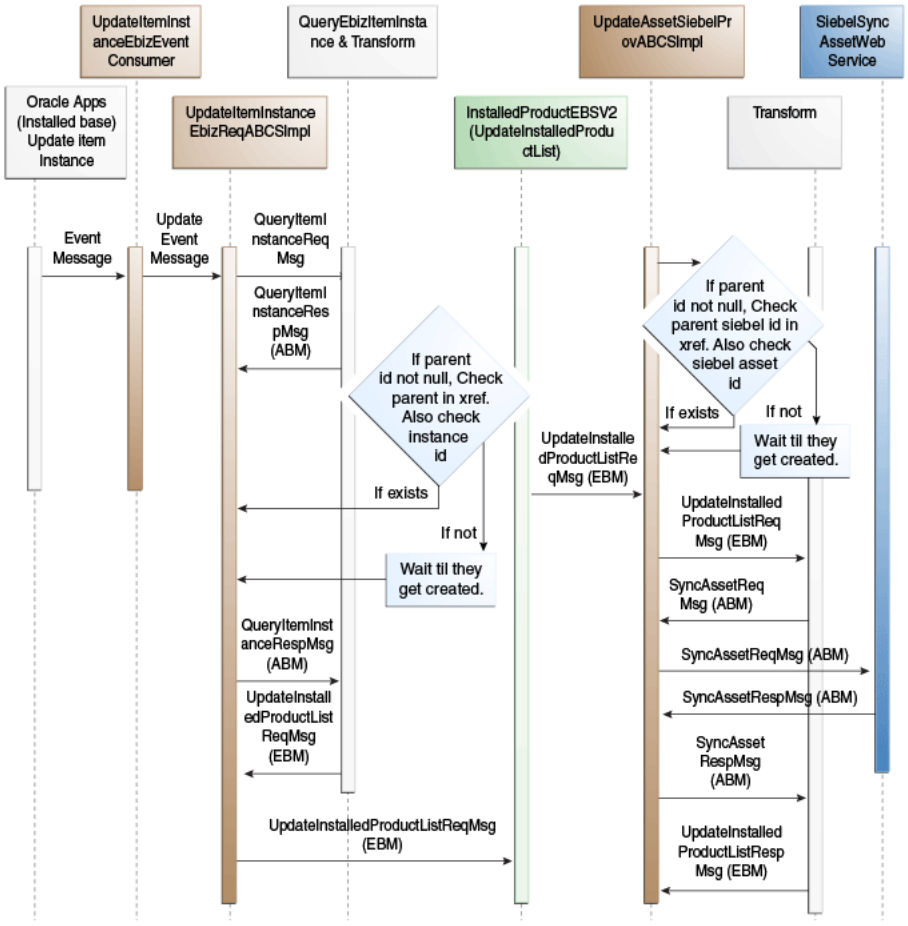
12.3.1 Update Assets Integration Flow

This integration flow uses the following interfaces:

- UpdateItemInstanceEbizEventConsumer
- QueryItemInstanceEbizAdapter
- UpdateItemInstanceEbizReqABCSImpl
- InstalledProductEBSV2
- UpdateAssetSiebelProvABCSImpl

Figure 12–6 illustrates the Update Assets integration scenario.

Figure 12–6 Update Assets Flow Sequence Diagram



When you initiate the Update Assets process, the following events occur:

1. The Oracle EBS update event is raised when an item instance is updated in Oracle Apps through the user interface, the Order fulfillment process, the RMA process, or when the Update Assets process is building relationships to represent a BOM structure.

   The event is raised for simple and complex asset updates. The UpdateItemInstanceEbizEventConsumer dequeues the message using the EBS adapter listening on the Update item instance event. The event payload is then routed to the UpdateItemInstanceEbizReqABCSImpl service.

2. The UpdateItemInstanceEbizReqABCSImpl service, using EBS adapter, queries the CSI_ASSET_INTEGRATION_V, passes in the Instance ID, and gets the entire item instance ABM message as QueryItemInstanceRespMsg.

3. The UpdateItemInstanceEbizReqABCSImpl then checks the cross-reference for the existence of the Parent ID.

   If the QueryItemInstanceRespMsg has the Parent ID for the item instance, then the UpdateItemInstanceEbizReqABCSImpl checks the cross-reference for the common Parent ID and the current Common Instance ID. This verification ensures that the Update process occurs after the Create process.

4. If the cross-reference does not exist for the parent, the UpdateItemInstanceEbizReqABCSImpl waits.

If the cross-reference does not exist for the parent or for the instance, the UpdateItemInstanceEbizReqABCSImpl waits for the parent and the instance that is to be created by the Create process, which must be initiated by the Create event from Oracle EBS. After it is created, the UpdateItemInstanceEbizReqABCSImpl continues with the rest of the processing, as described in the remaining steps.

5. If the cross-reference exists for the parent and the instance, the UpdateItemInstanceEbizReqABCSImpl transforms the QueryItemInstanceRespMsg into the UpdateInstalledProductEBM and then invokes the InstalledProductEBSV2 with the operation UpdateInstalledProductList

   For complex assets, the Parent ID, if available, is mapped as part of the Update transformation, and the relationship with the parent item instance is built.

6. The InstalledProductEBSV2 routes the EBM message to UpdateAssetSiebelProvABCSImpl.

7. The UpdateAssetSiebelProvABCSImpl checks for the existence of Siebel IDs in the cross-reference for the parent and the current instance, and then it transforms the UpdateInstalledProductEBM into the Siebel Asset ABM message CRMIntegSEBLHORAssetInterface as UpdateAssetReqMsg.

   Then it calls the Siebel Asset web service CRMIntegSEBLHORAssetInterface to update the asset in Siebel and returns a response message UpdateAssetRespMsg. The UpdateAssetSiebelProvABCSImpl then transforms the Siebel response message to the UpdateInstalledProductRespMsg, and then it sends the message back to InstalledProductEBSV2.

12.4 Siebel CRM Interfaces

The Siebel CRM interface for the CreateInstalledProduct flow and the UpdateInstalledProduct flow is SWIAssetManagementIO.

The CreateInstalledProduct flow and the UpdateInstalledProduct flow include the following services:

- SWIAssetManagement
- Request and Response Schema:
- SWIAssetManagementIO.xsd

For more information about Siebel web services, see *CRM Web Services Reference*.

12.5 Oracle EBS Interfaces

The Oracle EBS interface for the CreateInstalledProduct flow and the UpdateInstalledProduct flow is CSI_ASSET_INTEGRATION_V.

For more information about Oracle E-Business Suite web services and documentation prior to Release 12.1.3, see the library on Oracle Technology Network: http://www.oracle.com/technetwork/documentation/applications-167706.html?. For Oracle E-Business Suite documentation for R12.1.3 and beyond, see this library: http://download.oracle.com/docs/cd/E18727_01/index.htm?

12.6 Core Oracle AIA Components

The assets integration uses the following delivered core components:

- InstalledProductEBO
- CreateInstalledProductListEBM
- UpdateInstalledProductListEBM

The industry enterprise business object (EBO) and enterprise business message XML schema (EBM XSD) files are located here: $AIA_HOME/AIAMetaData/AIAComponents/EnterpriseObjectLibrary/Core/EBO/

The industry enterprise business service (EBS) WSDL files are located here: $AIA_HOME/AIAMetaData/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/

For detailed documentation of individual EBOs and EBMs, click the AIA Reference Doc link on the EBO and EBM detail pages in Oracle Enterprise Repository.

For more information about using the Oracle Enterprise Repository and configuring it to provide the AIA Reference Doc link, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring and Using Oracle Enterprise Repository as the Oracle AIA SOA Repository."

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

For more information, see *Oracle Fusion Middleware Concepts and Technologies Guide for Oracle Application Integration Architecture Foundation Pack*, "Understanding Extensibility."

12.7 Integration Services

The following services are delivered with the process integration for Asset Management:

- InstalledProductEBSV2
- QueryItemInstanceEbizAdapter
- QueryItemInstanceEbizR12VersionAdapter
- CreateItemInstanceEbizEventConsumer
- CreateItemInstanceEbizReqABCSImpl
- CreateAssetSiebelProvABCSImpl
- UpdateItemInstanceEbizEventConsumer
- UpdateItemInstanceEbizReqABCSImpl
- UpdateAssetSiebelProvABCSImpl

For more information, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring and Using Oracle Enterprise Repository as the Oracle AIA SOA Repository."

12.7.1 InstalledProductEBSV2

The InstalledProductEBSV2 is a routing service that exposes all of the enterprise operations that can be performed with the InstalledProductEBO. All of the integration flows for the Order to Cash integration make use of the operations provided by this enterprise business service.

The following operations on the InstalledProductEBSV2 are used in the asset integration flows:

- CreateInstalledProductList
- UpdateInstalledProductList

12.7.2 QueryItemInstanceEbizAdapter

The QueryItemInstanceEbizAdapter is a Mediator routing service in which the database adapter for querying the item instance is registered so that the database adapter call from the ABCSReqImpl becomes a Simple Object Access Protocol (SOAP) web service call.

12.7.3 QueryItemInstanceEbizR12VersionAdapter

The QueryItemInstanceEbizR12VersionAdapter is an adapter service. This queries the R12 Ebusiness suite view for an item instance. This adapter call from the ABCSReqImpl becomes a SOAP web service call. This service is specific for an R12 Ebusiness suite instance.

12.7.4 CreateItemInstanceEbizEventConsumer

The CreateItemInstanceEbizEventConsumer is a Mediator routing service that listens on the Create Item Instance event in Oracle Apps. It dequeues the event message and routes it to the CreateItemInstanceEbizReqABCSImpl service.

12.7.5 CreateItemInstanceEbizReqABCSImpl

The CreateItemInstanceEbizReqABCSImpl is a Business Process Execution Language (BPEL) process invoked by the CreateItemInstanceEbizEventConsumer service passing in the Create Item Instance payload.

This service queries the entire Item Instance Oracle EBS ABM message, then transforms into an EBM, and invokes the CreateInstalledProductList operation of the InstalledProductEBSV2.

This transformation is required: Oracle EBS Item Instance ABM to CreateInstalledProductListEBM.

12.7.6 CreateAssetSiebelProvABCSImpl

The CreateAssetSiebelProvABCSImpl is a BPEL process that is invoked by the CreateInstalledProduct operation of the InstalledProductEBSV2, passing in the Create Installed Product EBM payload. This service transforms the EBM into a Siebel ABM and invokes the create operation on the Siebel Asset web service interface.

The following transformations are required:

- Installed ProductList EBM to CreateAsset ABM
- CreateAsset response ABM to CreateInstalledProductList response EBM

12.7.7 UpdateItemInstanceEbizEventConsumer

The UpdateItemInstanceEbizEventConsumer is a Mediator routing service that listens on the Update Item Instance event in Oracle Apps, then dequeues the event message and routes it to the UpdateItemInstanceEbizReqABCSImpl service.

12.7.8 UpdateItemInstanceEbizReqABCSImpl

The UpdateItemInstanceEbizReqABCSImpl is a BPEL process that is invoked by the UpdateItemInstanceEbizEventConsumer service, passing in the Update Item Instance payload. This service queries the entire Item Instance Oracle EBS ABM message, transforms into an EBM, and invokes the UpdateInstalledProduct operation of the InstalledProductEBSV2.

This transformation is required: Ebiz Item Instance ABM to UpdateInstalledProductEBM.

12.7.9 UpdateAssetSiebelProvABCSImpl

The UpdateAssetSiebelProvABCSImpl is a BPEL process that is invoked by the UpdateInstalledProduct operation of the InstalledProductEBSV2, passing in the update Installed Product EBM payload.

This service transforms the EBM into the Siebel ABM and invokes the update operation on the Siebel Asset web service interface.

Part II

Implementing the Delivered Integrations

This part discusses prerequisites and data requirements, how to run initial data loads, and how to configure the Order to Cash pre-built integration.

Part II contains the following chapters:

13

Reviewing Prerequisites and Data Requirements

This chapter provides prerequisites and data requirements for initial data loads and each of the process integration flows.

This chapter includes the following sections:

13.1 Initial Data Loads: Prerequisites

Before proceeding with the data loads, ensure that:

- The Siebel CRM Integration Pack for Oracle Order Management pre-built integration is installed.
- Oracle Data Integrator (ODI) is installed on the same computer as AIA_HOME.
- ODI master and work repositories are created.
- Siebel is installed on Oracle, and users can access the Siebel base tables and Enterprise Integration Manager (EIM) tables for account, product, asset, and price list.
- If there are any schema changes on the Siebel side for mandatory columns, then they must be included in the sample ifb files provided so that the EIM jobs on the Siebel side run successfully.

- A database sequence with the name AIA_BULK_LOAD_SEQ is created in the target Siebel database schema. Use this SQL command to create the sequence CREATE or REPLACE sequence AIA_BULK_LOAD_SEQ.
- A database sequence with the name AIA_BULK_LOAD_SEQ is created in the intermittent target Xref database schema. Use this SQL command to create the sequence CREATE sequence AIA_BULK_LOAD_SEQ
- For pricelist initial load, QP_CRMINTEG_PRICELIST_V retrieves records for the pricing organization setup in the profile option QP: Item Validation Organization at the site level. This profile option must be setup with the organization for which pricing data has to be retrieved.

13.2 Customer Management: Prerequisites and Data Requirements

This section discusses the prerequisites and data requirements for the Customer Management process integration flow.

13.2.1 Prerequisites

The process integration for customer management does not depend on other processes being run; however, the Organization cross-reference must be set up first.

For more information about setting up cross-references, see Chapter 15, "Configuring the Order to Cash Pre-Built Integration."

13.2.2 Data Requirements

The process integration for customer management data requirements are:

- The Siebel Customer Relationship Management (Siebel CRM) business units and Oracle E-Business Suite (Oracle EBS) operating units must be mapped as described in Chapter 15, "Configuring the Order to Cash Pre-Built Integration."
- Oracle EBS profile options must be specified.
- The contact associated with the order must have the same account associated with it.
- Specify the contact name on the Payments tab of the Sales Order screen.
- The address must include address line 1, city, state, and zip code.
- Different organization parties cannot share locations in Oracle EBS, though they can share addresses between accounts in Siebel CRM. Therefore, the address should be re-created in Siebel CRM.
- In Oracle EBS, the contact should be associated at the account level only. For the Order Management application programming interface (API) to process the contact, it should not be associated at multiple levels of an account. For example, the contact should not be associated at the account site (address) level.
- If the contact used in Siebel CRM during Order submit is associated with multiple accounts and contact points (phone numbers or email address) are present, then Order submit is not supported because the Oracle EBS API for contact update or create does not support shared contact points.

For more information about setting up Siebel organizations and about Oracle EBS operating units and profile options, see Chapter 15, "Configuring the Order to Cash Pre-Built Integration."

13.3 Product Management: Prerequisites and Data Requirements

This section discusses the prerequisites and data requirements for the Product Management process integration flow.

13.3.1 Prerequisites

The process integration for product management does not have a dependency on other processes; however, the following steps should be performed:

- Set up organization cross-references.
- Set up inventory location cross-references.

For more information about setting up the organizations and cross-references, see Chapter 15, "Configuring the Order to Cash Pre-Built Integration."

13.3.2 Data Requirements

The process integration for product management has the following data requirements:

1. The Item Name field in Oracle EBS consists of concatenated key flexfield segments. It cannot exceed 50 characters in length.
2. Do not use special characters, such as *&*, in the Oracle EBS item name field definition.
3. For an Item to be synchronizable from Oracle EBS:
   - It must have a Customer Orderable Flag
   - The Item Type value must be either *Model, Option Class*, or *Standard*
   - The Item must belong to an Oracle EBS Item Validation Org.
4. The unique key to product in Siebel CRM is product name, organization (business unit), and vendor account (not mapped in this integration). The organization name in Siebel CRM must be unique. Siebel CRM supports duplicate product names across organizations but not within an organization.

13.4 Price Lists: Prerequisites

This section discusses prerequisites for the Price List process integration flow.

13.4.1 Prerequisites

The prerequisite for the process integration for price lists is to run the product synchronization flow.

For more information about product synchronization, see Chapter 4, "Process Integration for Product Management."

13.5 Quotes: Prerequisites and Data Requirements

This section discusses prerequisites and data requirements for the Quotes process integration flow.

13.5.1 Prerequisites

The prerequisites for the process integration for quotes integration are:

- Product synchronization.
- Pricing synchronization.

For more information, see Chapter 4, "Process Integration for Product Management." and Chapter 5, "Process Integration for Price Lists."

13.5.2 Data Requirements

This list indicates the mandatory information that must be provided to make this flow successful:

- The Quotes integration flow must have a minimum of one quote line item, and the quote line item must have a product synchronized from the back office.
- The quote must have valid customer information details, including account, billing and shipping address, and contact information.

13.6 Available to Promise Check: Prerequisites and Data Requirements

This section discusses prerequisites and data requirements for the Available to Promise (ATP) Check process integration flow.

13.6.1 Prerequisites

The prerequisite for the ATP Check integration flow is product synchronization.

For more information about the product synchronization flow, see Chapter 4, "Process Integration for Product Management."

13.6.2 Data Requirements

This list indicates the mandatory information that must be provided to make this flow successful:

- Order has at least one line item.
- The quantity on the order line is greater than zero.
- A product that has been synchronized from Oracle EBS is specified.
- A valid requested delivery date is specified.

13.7 Shipping Charges: Prerequisites and Data Requirements

This section discusses prerequisites and data requirements for the Shipping Charges process integration flow.

13.7.1 Prerequisites

The prerequisites for the shipping charges integration flow are:

1. The `glog.integration.remoteQuery.wrapReplyInTransmission` property must be set for OTM 5.5 CU5.

   For more information about setting this property, see Section 15.12, "Setting a Property in OTM."

2. Product synchronization.

For more information about the product synchronization flow, see Chapter 4, "Process Integration for Product Management."

13.7.2 Data Requirements

The data requirements for the shipping charges integration flow are:

- Order has at least one line item.
- The order line item has a valid product.
- The product weight is specified.
- The source and target addresses are specified.
- The source inventory location is specified.

13.8 Credit Check: Prerequisites and Data Requirements

This section discusses prerequisites and data requirements for the Credit Check process integration flow.

13.8.1 Prerequisites

The prerequisite for the Credit Check integration flow is product synchronization.

For more information about product synchronization, see Chapter 4, "Process Integration for Product Management."

13.8.2 Data Requirements

This list indicates the mandatory information that must be provided to make this flow successful:

- Payment method is Purchase Order.
- Order has account information.
- Transaction amount is greater than zero.
- Order has at least one line item.
- Order quantity is greater than zero.

13.9 Payment Authorization: Prerequisites and Data Requirements

This section discusses perquisites and data requirements for the Payment Authorization process integration flow.

13.9.1 Prerequisites

For the Payment Authorization flow to work, the iPayment application must be configured properly.

For more information about iPayment, see the product documentation.

13.9.2 Data Requirements

This list indicates the mandatory information that must be provided to make the flow successful:

- Valid credit card data.
- Order has one line item, and quantity is greater than zero.
- Transaction amount is greater than zero.

13.10 Order Management: Prerequisites and Data Requirements

This section discusses prerequisites and data requirements for the Order Management process integration flow.

13.10.1 Prerequisites

The prerequisites for the process integration for order management are:

- Product synchronization.
- Account synchronization.
- Price List synchronization (optional).
- Organization cross-reference setup.

For more information about these process integrations, see Chapter 13, "Reviewing Prerequisites and Data Requirements," Chapter 4, "Process Integration for Product Management," and Chapter 5, "Process Integration for Price Lists."

13.10.2 Data Requirements

The process integration for order management has the following data requirements:

1. The order must be of the type *Sales Order*.
2. The sales order must contain account, billing and shipping address, and shipping contact.
3. Orders are submitted with the *Booked* status.
4. If Price List integration is implemented, the price list name must be specified on the order header.
5. Only one payment type can be used for the entire order.
6. If an order line is revised or canceled, a change reason value must be provided at the line-level. The default value is *No Reason*.
7. If the entire order is being canceled, the cancel reason must be provided at the order header-level.

13.11 Asset Management: Prerequisites and Data Requirements

This section discusses prerequisites and data requirements for the Asset Management process integration flow.

13.11.1 Prerequisites

The cross-business process functional prerequisites for asset integration are:

- Product synchronization.
- Customer Account synchronization.

For more information about these process integrations, see Chapter 3, "Process Integration for Customer Management" and Chapter 4, "Process Integration for Product Management."

13.11.2 Data Requirements

Product and customer data must be synchronized for the asset integration flow to work properly.

14

Running Initial Data Loads

This chapter provides step-by-step instructions on how to configure the Oracle Data Integrator (ODI) and then perform initial data loads for customer data, product data, price list data, and assets data.

This chapter includes the following sections:

To see the prerequisites for running initial data loads, see Chapter 13, "Reviewing Prerequisites and Data Requirements."

14.1 Deploying ODI Repository Components

Complete the following steps to deploy the Order to Cash ODI repository components.

> **Note:** After the scripts run successfully, log in to the ODI Topology Manager and Designer to verify that all of the components loaded successfully. Also, test to ensure that the data server connection works.

14.1.1 Configuring ODI Details

This section can be bypassed if ODI details were provided when configuring the Order to Cash: Siebel CRM - EBS integration. Otherwise, complete the following steps to provide the ODI installation and connection details.

For more information about configuring the Order to Cash: Siebel CRM - EBS integration, see the *Oracle Application Integration Architecture Installation and Upgrade Guide for Pre-Built Integrations*.

To provide the ODI installation and connection details:

1. To understand the ODI setup related prerequisites, see the *Oracle Application Integration Architecture Installation and Upgrade Guide for Pre-Built Integrations*, "Configuring and Deploying the Order to Cash: Siebel CRM - EBS Integration", Configuring ODI (Optional).

2. Navigate to <AIA_Instance>/bin and run the following command to configure the installation environment:
   - For Windows: aiaenv.bat
   - For Linux: aiaenv.sh
3. Navigate to <AIA_HOME>/bin and run the following command to launch the Oracle AIA Configuration Wizard:
   - For Windows: aiaconfig.bat
   - For Linux: ./aiaconfig.sh
4. Click **Next**.
5. Select the Order to Cash: Siebel CRM - EBS Initial Loads. Click **Next**.
6. Specify ODI access details, master repository details, and work repository for Order to Cash details.

For more information and description of the fields for ODI related configuration screens, see the *Oracle Application Integration Architecture Installation and Upgrade Guide for Pre-Built Integrations*, "Configuring and Deploying the Order to Cash: Siebel CRM - EBS Pre-Built Integration", Deployment Configuration Wizard Interview.

See the sections related to the following screens:

- ODI Access Details Screen (Optional)
- ODI Master Repository Details Screen (Optional)
- ODI Work Repository Details for Order to Cash Screen (Optional)

14.1.2 In UNIX

Set the environment variables using:

```
Source: aiaenv.sh from AIA_INSTANCE/bin
```

To deploy components for individual bulk loads:

For Customer, run:

```
ant -f $AIA_HOME/Infrastructure/Install/AID/AIAInstallDriver.xml
-DPropertiesFile=$AIA_INSTANCE/config/AIAInstallProperties.xml
-DDeploymentPlan=$AIA
_HOME/services/core/BulkDataProcess/EbizToSiebel/Customer/
O2C_ODI_Customer_InstallScript.xml
```

For Product, run:

```
ant -f $AIA_HOME/Infrastructure/Install/AID/AIAInstallDriver.xml
-DPropertiesFile=$AIA_INSTANCE/config/AIAInstallProperties.xml
-DDeploymentPlan=$AIA
_HOME/services/core/BulkDataProcess/EbizToSiebel/Product/
O2C_ODI_Product_InstallScript.xml
```

For Price List, run:

```
ant -f $AIA_HOME/Infrastructure/Install/AID/AIAInstallDriver.xml
-DPropertiesFile=$AIA_INSTANCE/config/AIAInstallProperties.xml
-DDeploymentPlan=$AIA
_HOME/services/core/BulkDataProcess/EbizToSiebel/PriceList/
O2C_ODI_PriceList_InstallScript.xml
```

For Asset, run:

```
ant -f $AIA_HOME/Infrastructure/Install/AID/AIAInstallDriver.xml
-DPropertiesFile=$AIA_INSTANCE/config/AIAInstallProperties.xml
 -DDeploymentPlan=$AIA
_HOME/services/core/BulkDataProcess/EbizToSiebel/Assets/
O2C_ODI_Asset_InstallScript.xml
```

14.1.3 In Windows

Set environment as:

```
aiaenv.bat from AIA_INSTANCE\bin
```

To deploy components for individual bulk loads:

For Customer, run:

```
ant -f %AIA_HOME%\Infrastructure\Install\AID\AIAInstallDriver.xml
-DPropertiesFile=%AIA_INSTANCE%\config\AIAInstallProperties.xml
-DDeploymentPlan=%AIA
_HOME%\services\core\BulkDataProcess\EbizToSiebel\Customer\
O2C_ODI_Customer_InstallScript.xml
```

For Product, run:

```
ant -f %AIA_HOME%\Infrastructure\Install\AID/AIAInstallDriver.xml
-DPropertiesFile=%AIA_INSTANCE%\config\AIAInstallProperties.xml
-DDeploymentPlan=%AIA
_HOME%\services\core\BulkDataProcess\EbizToSiebel\Product\
O2C_ODI_Product_InstallScript.xml
```

For Price List, run:

```
ant -f %AIA_HOME%\Infrastructure\Install\AID\AIAInstallDriver.xml
-DPropertiesFile=%AIA_INSTANCE%\config\AIAInstallProperties.xml
-DDeploymentPlan=%AIA
_HOME%\services\core\BulkDataProcess\EbizToSiebel\PriceList\
O2C_ODI_PriceList_InstallScript.xml
```

For Asset, run:

```
ant -f %AIA_HOME%\Infrastructure\Install\AID\AIAInstallDriver.xml
-DPropertiesFile=%AIA_INSTANCE%\config\AIAInstallProperties.xml
-DDeploymentPlan=%AIA
_HOME%\services\core\BulkDataProcess\EbizToSiebel\Assets\
O2C_ODI_Asset_InstallScript.xml
```

14.1.4 Setting Up a Data Server for a Non-Oracle Database

For more information about how to set up a data server for a non-Oracle database, see Appendix A, "Configuring ODI-Based Initial Loads against a Non-Oracle Target Database."

14.2 Manual Changes for Oracle Data Integrator

The following steps are manual changes you must make to the Order to Cash ODI components for ODI version 11.1.1.5.

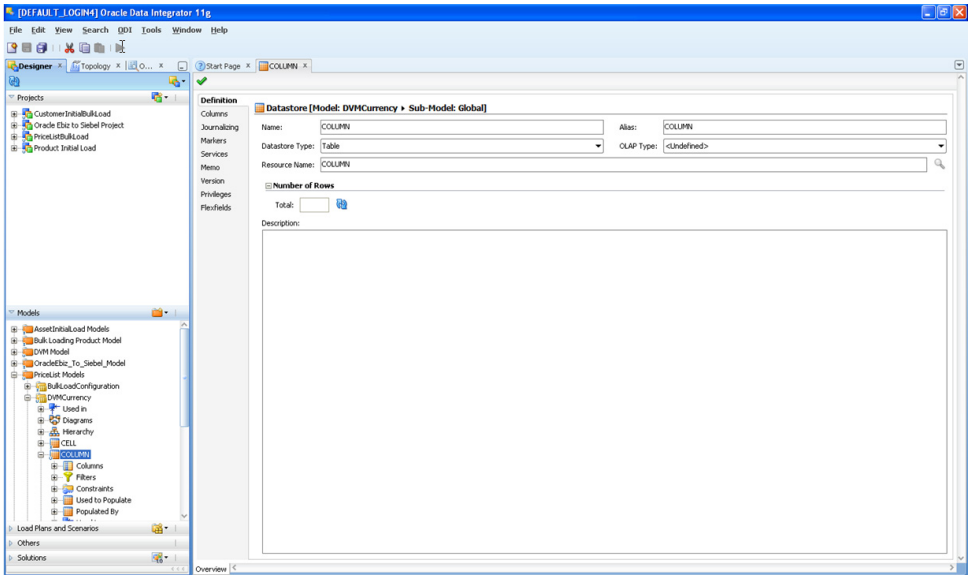
In ODI 11.1.1.5.0, after the components are deployed, log in to ODI designer and change the ODI Models and then the ODI Interfaces. These changes are textual changes to attribute names only.

14.2.1 Changing the ODI Models

Perform these steps:

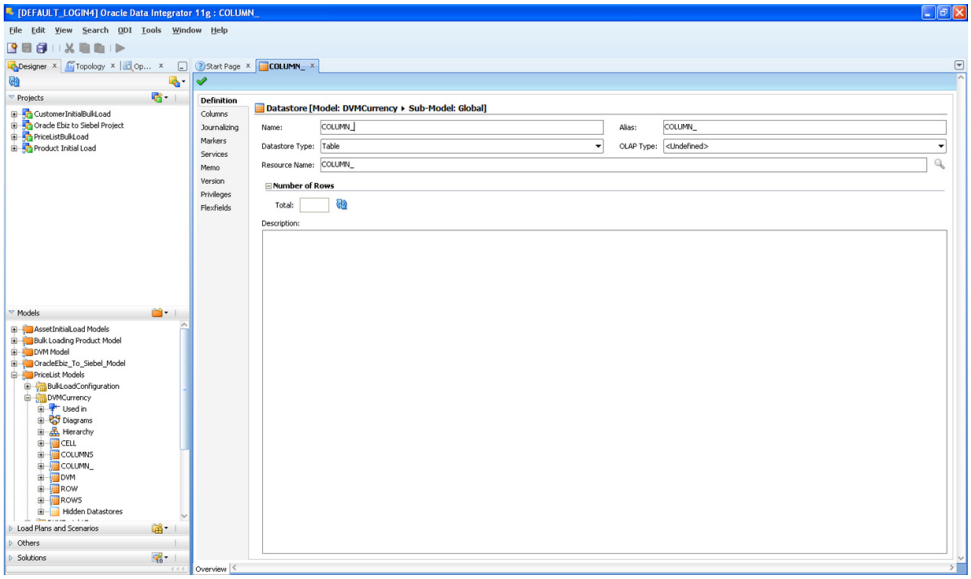
1. Click the + sign to open the Models one by one.
2. Double click the **Column** attribute to open the Definition page as shown in Figure 14–1. Not all models have a Column attribute. Only those models that have a Column attribute must be modified.

Figure 14–1 Column Definition page



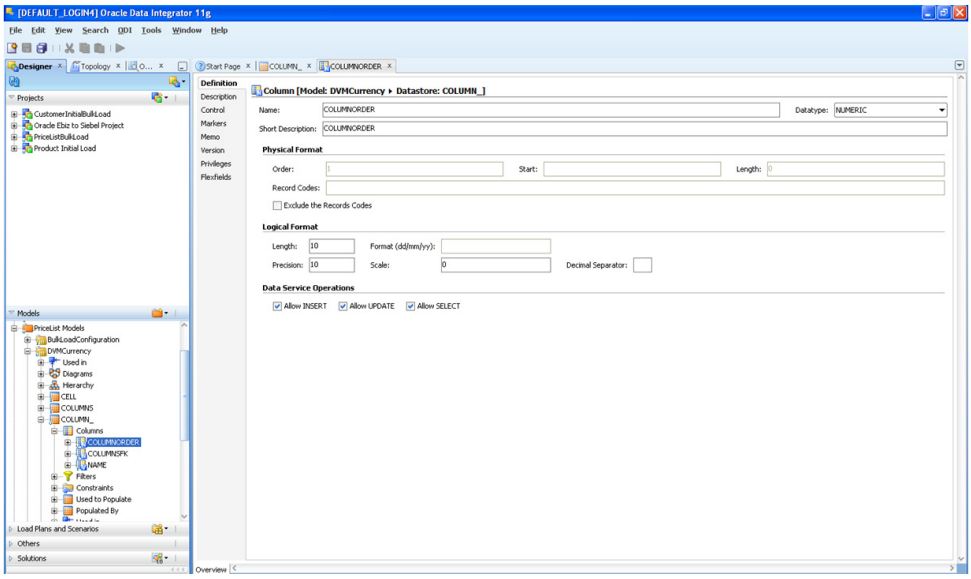
3. In the Definition page, shown in Figure 14–2, change the **Name, Alias**, and **Resource Name** from *Column* to *Column_* and save.

Figure 14–2 Column Definition page



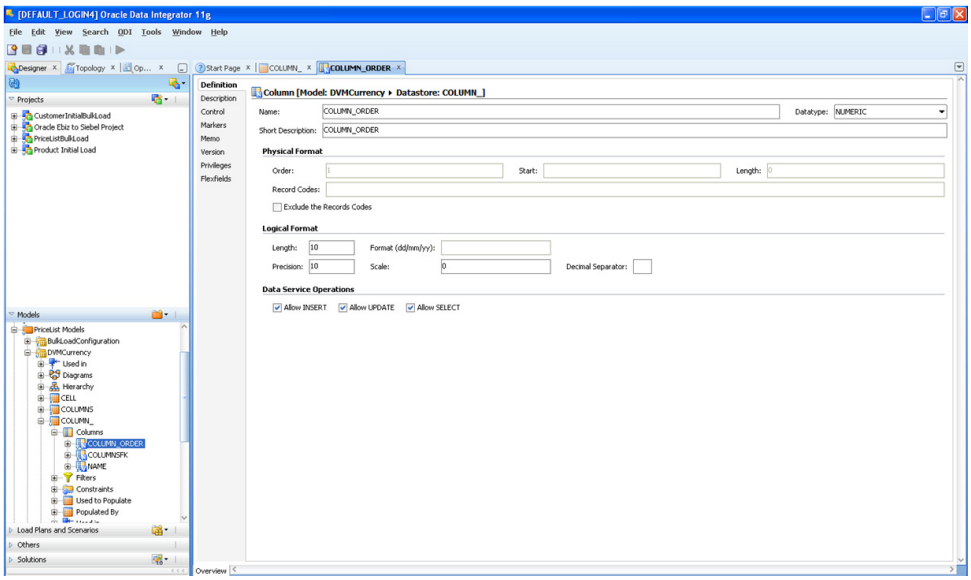
4. Next, click + to open *Column_* click + on *Columns*, and then open COLUMNORDER as shown in Figure 14–3.

Figure 14–3 COLUMNORDER page



5. In the Definition page shown in Figure 14–4, change the **Name** and **Short Description** from COLUMNORDER to COLUMN_ORDER and click **Save**.

Figure 14–4 Definition page



6. Repeat these steps for all Models that have a Column attribute.

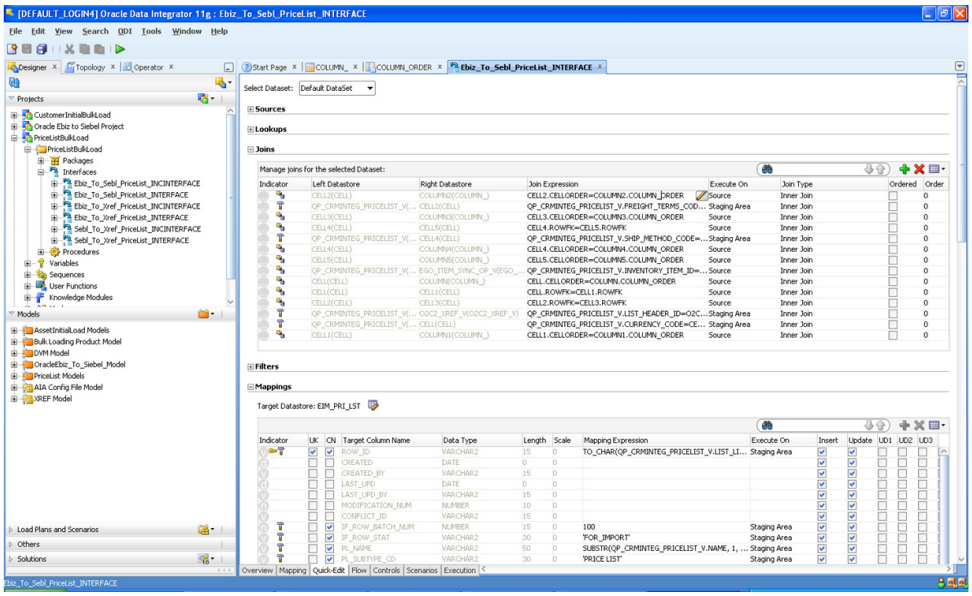
14.2.2 Changing the ODI Interfaces

Perform the following steps:

1. Under each of the projects, open the interfaces one by one.
2. In the interface, click the **Quick-Edit** tab at the bottom of the screen.
3. Click + sign next to Joins to open the joins.

4. In the **Join Expression**, shown in Figure 14–5 change all of the COLUMNORDER to COLUMN_ORDER and click **Save**.

Figure 14–5 Join Expressions



5. Repeat these steps for all interfaces in all projects.
6. Finally, regenerate all of the scenarios.

14.2.3 Changing the XML DataSource

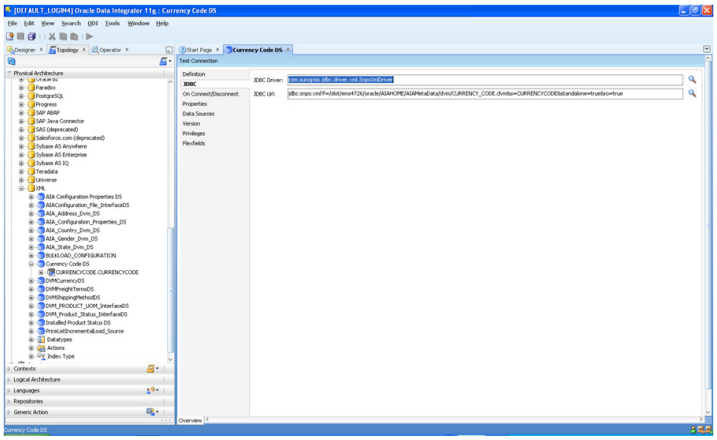
Perform these steps:

1. Select **Topology** Tab from the ODI Menu.
2. Click the **+** sign of **Technologies** to expand it.
3. Click the **+** sign of **XML** folder.
4. Select *CurrencyCodeDS*, right click, and select **Open**.
5. Click the **JDBC** tab to view the JDBC URL.
6. Check the value in the **JDBC URL** and append the following string `&ro=true` to the end of the URL. Ensure that this is not present in the URL.
7. The JDBC URL for this DataSource should look like this:

```
Xxxx/oracle/AIAHome/AIAMetaData/dvm/CURRENCY_
CODE.dvm&s=CURRENCYCODE&standalone=true&ro=true
```

Figure 14–6 shows the Currency Code DS screen:

Figure 14–6 Currency Code DS



14.3 Loading Initial Customer Data

Siebel CRM Integration Pack for Oracle Order Management provides a bulk load feature to move data from Oracle E-Business Suite (Oracle EBS) into Siebel Customer Relationship Management (Siebel CRM).

The bulk load feature extracts customer data from Oracle EBS, transforms it into Siebel Enterprise Integration Manager (Siebel EIM) interface table data structures, and moves it into the Siebel EIM interface tables. Siebel EIM tables are intermediate interface tables that act as staging areas between the base tables in the Siebel database and other databases. A Siebel loader program moves the data from the EIM interface tables into the application base tables of Siebel CRM.

During the initial bulk load, only active organization parties and their child objects, including organization contacts that are associated with an account in Oracle, are extracted from Oracle and loaded into Siebel CRM. Organization contacts are person parties that have a relationship with the organization party.

While Oracle organization parties are loaded into Siebel CRM as accounts, Oracle organization contacts are loaded into Siebel CRM as account contacts. (Similarly, a contact record in Siebel CRM is loaded for a contact person party in Oracle EBS).

Note: Person parties and their child objects are not loaded because Siebel CRM Integration Pack for Oracle Order Management does not support a B2C (business to customer) ordering flow.

14.3.1 Loading Customer and Contact Bulk Data

The Customer bulk load consists of these steps:

1. Load customer data from Oracle EBS to Siebel EIM

   Use ODI to populate the Siebel EIM table with data from the Oracle EBS table. Account, address, and contact entities are loaded into the respective EIM tables. The Oracle Application Integration Architecture (Oracle AIA) cross-reference table is populated with the Oracle EBS and Common ID values.

2. Load EIM values to Siebel base tables.

   Use an EIM job to transfer the data from EIM to the Siebel base tables.

3. Load Siebel base tables to the Oracle AIA cross-reference table.

Use ODI to move the Siebel ID values back into the Oracle AIA cross-reference table, correlating to the Oracle EBS and Common ID values that were loaded in step 1.

For initial data loads, you must perform some EIM activities.

For more information about the pre- and post-EIM activities and execution, see the *Siebel Enterprise Integration Manager Administration Guide*, Siebel EIM Tables.

14.3.1.1 Moving Data from Oracle EBS to Siebel EIM Tables

You can move data from Oracle EBS to Siebel EIM tables using either ODI Designer or a command line prompt.

To move data from Oracle EBS using ODI designer:

1. Log in to the Designer.
2. Expand the project **CustomerInitialBulkLoad**.
3. Expand the folder **Customer Bulk Load**.
4. Expand the package.
5. Right-click the package titled `OracleEbiz_to_XREF_to_EIM` and select **Execute** from the menu.
6. Select *My_Context* in the **Execution** box.

To move data from Oracle EBS using a command line prompt:

1. Go to the bin folder of ODI HOME in the Command window.
2. Enter the following command:
   - For Windows: startscen ORACLEEBIZ_TO_XREF_TO_EIM 001 MY_CONTEXT
   - For UNIX: ./startscen.sh ORACLEEBIZ_TO_XREF_TO_EIM 001 MY_CONTEXT

14.3.1.2 Moving Data from EIM to Siebel Base Tables

To move data from EIM to Siebel base tables:

1. Verify that the Product.ifb, Pricelist.ifb, Customer.ifb, and Asset.ifb files are available in the Admin directory of the installed Siebel server.

   Example: in the D:\ses\siebsrvr\Admin directory

2. Modify Customer.ifb as follows and save.

   Under the DEFAULT COLUMN list for [IMPORT CONTACT], add the following:

   DEFAULT COLUMN = CON_ONDMNDSYNCFLG, "N"

   Under the DEFAULT COLUMN list for [IMPORT ADDRESS], add the following:

   DEFAULT COLUMN = ADDR_GEOCD_VLD_FLG,"N"

   DEFAULT COLUMN = ADDR_SYNC_SUCC_FLG,"N"

   DEFAULT COLUMN = ADDR_VERIFIED_FLG,"N"

   Under [UPDATE ACCOUNT NUMBER], ensure that the database user name from the Siebel environment is prefixed to the table name. Here is an example:

SESSION SQL = "UPDATE ORA05223.EIM_ACCOUNT SET OU_NUM = T_ORG_EXT__RID WHERE IF_ROW_BATCH_NUM=100"

3. Log in to the Siebel E-Business Applications using the Administrator login.

   Example: proper credentials like SADMIN/SADMIN

4. Go to the **Site Map** icon (the globe that appears at the top left of the home page).
5. Navigate to the Administration - Server Management screen.
6. Click the **Jobs** link and navigate to **Server Management, Jobs**.
7. Click **New** in the Jobs applet.
8. Ensure that you set the component to Enterprise Integration Manager in this new job.
9. Go to **Job Parameters** and enter the following parameter values:
   - Configuration File: Customer.ifb
   - Error Flag: 1
   - SQL Trace Flag: 8
   - Trace Flag: 1
10. Ensure that the job finishes with a status of *Success*.

    The color of the status bar changes to green.

11. Ensure that the data appears properly in the Siebel Business Applications User Interface in relevant Administration screens, such as Accounts, Contacts, Products, Price List, and so on.
12. To check whether the Siebel EIM loaded all records, check the IF_ROW_STAT column for the corresponding EIM table.

    The IF_ROW_STAT field can have following status values:
    - IF_ROW_STAT = for import: Job to be run or job running.
    - IF_ROW_STAT = Partially_imported: Some mandatory field is missing for that record.
    - IF_ROW_STAT = Imported: Job ran successfully for that record.
    - IF_ROW_STAT= dup_row_exists: Siebel base table has that record.

14.3.1.3 Moving Data from Siebel Base Tables to Oracle AIA Cross-Reference Tables

You can move data from Siebel base tables to Oracle AIA cross-reference tables using either ODI Designer or a command line prompt.

To move data from Siebel base tables using ODI Designer:

1. Log in to the Designer.
2. Expand the project **CustomerInitialBulkLoad**.
3. Expand the folder **Customer_Bulk_Load**.
4. Expand the package.
5. Right-click the package titled `Siebel_to_XREF` and select **Execute** from the menu.

6. Select *My_Context i*n the Execution box.

To move data from Oracle EBS using a command line prompt:

1. Go to the bin folder of ODI HOME in the Command window.
2. Enter the following command:
   - For Windows: Startscen SIEBEL_TO_XREF 001 MY_CONTEXT
   - For UNIX:./startscen.sh SIEBEL_TO_XREF 001 MY_CONTEXT

14.3.1.4 Verifying Data After the Load

Use this table to verify the data:

Table 14–1 Information to verify Data After the Load

Entity	Database	Query	Inference
ACCOUNTS	EBIZ	select DISTINCT HZ_CUST_ACCOUNTS.CUST_ACCOUNT_ID from APPS.HZ_CUST_ACCOUNTS HZ_CUST_ACCOUNTS, APPS.HZ_PARTIES HZ_PARTIES, APPS.HZ_CUST_ACCT_SITES_ALL HZ_CUST_ACCT_SITES_ALL where (HZ_PARTIES.PARTY_TYPE='ORGANIZATION') And (NVL(HZ_CUST_ACCOUNTS.STATUS,'A')='A') And (HZ_PARTIES.STATUS='A') And (HZ_CUST_ACCOUNTS.CUST_ACCOUNT_ID=HZ_CUST_ACCT_SITES_ALL.CUST_ACCOUNT_ID) AND (HZ_PARTIES.PARTY_ID=HZ_CUST_ACCOUNTS.PARTY_ID);	Number of active accounts in Oracle EBS
ACCOUNTS	SIEBEL	select count(*) from s_org_ext where db_last_upd_src='EIM';	Number of accounts successfully transferred to Siebel from the EIM job.

Table 14–1 (Cont.) Information to verify Data After the Load

Entity	Database	Query	Inference
CONTACTS	EBIZ	Select DISTINCT HZ_PERSON_PROFILES.PARTY_ID C1_VALUE from APPS.HZ_PERSON_PROFILES HZ_PERSON_PROFILES, APPS.HZ_RELATIONSHIPS HZ_RELATIONSHIPS, APPS.HZ_CUST_ACCT_SITES_ALL HZ_CUST_ACCT_SITES_ALL, APPS.HZ_CUST_ACCOUNT_ROLES HZ_CUST_ACCOUNT_ROLES, APPS.HZ_CUST_ACCOUNTS HZ_CUST_ACCOUNTS, APPS.HZ_PARTIES HZ_PARTIES where (1=1) And (HZ_RELATIONSHIPS.OBJECT_TYPE='ORGANIZATION') And (HZ_RELATIONSHIPS.RELATIONSHIP_CODE='CONTACT_OF') And (HZ_RELATIONSHIPS.STATUS='A') And (HZ_CUST_ACCOUNTS.STATUS='A') And (HZ_PARTIES.STATUS='A') And (HZ_CUST_ACCOUNTS.PARTY_ID=HZ_PARTIES.PARTY_ID) AND (HZ_PERSON_PROFILES.PARTY_ID=HZ_RELATIONSHIPS.SUBJECT_ID) AND (HZ_RELATIONSHIPS.OBJECT_ID=HZ_CUST_ACCOUNTS.PARTY_ID) AND (HZ_RELATIONSHIPS.PARTY_ID=HZ_CUST_ACCOUNT_ROLES.PARTY_ID) AND (HZ_CUST_ACCOUNTS.CUST_ACCOUNT_ID=HZ_CUST_ACCT_SITES_ALL.CUST_ACCOUNT_ID) AND (HZ_CUST_ACCOUNT_ROLES.CUST_ACCOUNT_ID=HZ_CUST_ACCT_SITES_ALL.CUST_ACCOUNT_ID)	Number of contacts of the active accounts and parties.
CONTACTS	SIEBEL	select count(*) from s_contact where db_last_upd_src='EIM';	Number of contacts successfully transferred to Siebel from the EIM job.

Table 14–1 (Cont.) Information to verify Data After the Load

Entity	Database	Query	Inference
ADDRESS	EBIZ	select DISTINCT HZ_LOCATIONS.LOCATION_ID C1_ LOCATION_ID from APPS.HZ_LOCATIONS HZ_ LOCATIONS, APPS.HZ_PARTY_SITES HZ_PARTY_SITES, APPS.HZ_PARTIES HZ_PARTIES, APPS.HZ_CUST_ ACCOUNTS HZ_CUST_ACCOUNTS, APPS.HZ_CUST_ACCT_SITES_ALL HZ_CUST_ACCT_SITES_ALL where (1=1) And (HZ_PARTIES.PARTY_ TYPE='ORGANIZATION') And (HZ_PARTIES.STATUS='A') And (HZ_CUST_ ACCOUNTS.STATUS='A') And (HZ_PARTY_SITES.PARTY_ ID=HZ_PARTIES.PARTY_ID) AND (HZ_LOCATIONS.LOCATION_ ID=HZ_PARTY_SITES.LOCATION_ID) AND (HZ_CUST_ACCOUNTS.PARTY_ ID=HZ_PARTIES.PARTY_ID) AND (HZ_CUST_ACCOUNTS.CUST_ ACCOUNT_ID=HZ_CUST_ACCT_ SITES_ALL.CUST_ACCOUNT_ID)	Number of addresses of the active accounts and parties.
ADDRESS	SIEBEL	select count(*) from S_ADDR_PER where db_last_upd_src='EIM';	Number of addresses successfully transferred to Siebel from the EIM job.

14.4 Loading Initial Product Data

Use the bulk load feature to move product data from Oracle EBS into Siebel CRM.

Bulk loading product data is similar to bulk loading customer data. The bulk load feature extracts product data from Oracle EBS, transforms it into the EIM interface table data structures, and moves it into the Siebel EIM interface tables. A Siebel loader program moves the data from the EIM interface tables into the application base tables of Siebel CRM.

14.4.1 Loading Product Bulk Data

The Product bulk load consists of these steps:

1. Load Oracle EBS to EIM.

   Use ODI to populate the Siebel EIM table with data from the Oracle EBS table. The Oracle AIA cross-reference table is populated with the Oracle EBS and Common ID values.

2. Transfer EIM data to the Siebel Base table.

   Use an EIM job to transfer the data from EIM to the Siebel Base table.

3. Move values from the Siebel Base table to the Oracle AIA cross-reference table.

Use ODI to move the Siebel ID values back into the Oracle AIA cross-reference table, correlating to the Oracle EBS and Common ID values that were loaded in step 1.

For initial data loads, you must perform some EIM activities.

For more information about the pre- and post-EIM activities and execution, see the *Siebel Enterprise Integration Manager Administration Guide*, Siebel EIM Tables.

14.4.1.1 Moving Data from Oracle EBS to Siebel EIM Tables

You can move data from Oracle EBS to Siebel EIM tables using either ODI Designer or a command line prompt.

To move data from Oracle EBS using the ODI Designer:

1. Log in to the Designer.
2. Expand the project **Product Initial Load**.
3. Expand the folder **Bulk Load**.
4. Expand the package.
5. Right-click the package titled `Load_Oracle_Ebiz_Product_To_Siebel`. Select **Execute** from the menu.
6. Select the *Product Development* context in the execution box.

To move data from Oracle EBS using a command line prompt:

1. Go to the bin folder of ODI HOME in the Command window.
2. Enter the following command:
   - For Windows: Startscen LOAD_ORACLE_EBIZ_PRODUCT_TO_SIEBEL 001 PRODUCTDEVELOPMENT
   - For UNIX: ./startscen.sh LOAD_ORACLE_EBIZ_PRODUCT_TO_SIEBEL 001 PRODUCTDEVELOPMENT

14.4.1.2 Moving Data from EIM to Siebel Base Tables

To move data from EIM to Siebel base tables:

1. Verify that the Product.ifb, Pricelist.ifb, Customer.ifb, Asset.ifb files are available in the Admin directory of the installed Siebel server.

   For example, in the D:\ses\siebsrvr\Admin directory
2. Modify Product.ifb as follows and save:

   Under the DEFAULT COLUMN list for [IMPORT EIM INT], add the following:

   DEFAULT COLUMN = APPLY_MEM_COMP_FLG,"N"

   DEFAULT COLUMN = ONDEMAND_SYNC_FLG,"N"

   DEFAULT COLUMN = RSLV_AMBIGUITY_FLG,"N"
3. Log in to the Siebel E-Business Applications using the Administrator login.

   For example, use proper credentials such as SADMIN/SADMIN
4. Go to the **Site Map** icon (the globe that appears at the top left of the home page).
5. Navigate to the Administration - Server Management screen.

6. Click the **Jobs** link and navigate to **Server Management, Jobs**.
7. Click **New** in the Jobs applet.
8. Ensure that the component is set to Enterprise Integration Manager in this new job.
9. Go to **Job Parameters** and enter the following parameter values:
   - Configuration File: Product.ifb
   - Username: The administrator username ex. SADMIN
   - Password: The administrator password ex. SADMIN
   - Error Flag: 1
   - SQL Trace Flag: 8
   - Trace Flag: 1
10. Ensure that the job finishes with a status of *Success*.

    The color of the status bar changes to green.
11. Release the products after getting the product data imported into the base table.

    See the EIM documentation provided by Siebel.
12. Ensure that the data appears correctly in the Siebel Business Applications User Interface in the relevant Administration screens such as Accounts, Contacts, Products, Price List, and so on.
13. To check whether the Siebel EIM loaded all records, check the IF_ROW_STAT column for the corresponding EIM table.

    The IF_ROW_STAT field can have the following status values:
    - IF_ROW_STAT = for import: Job to be run or job running.
    - IF_ROW_STAT = Partially_imported: Some mandatory field missing for that record.
    - IF_ROW_STAT = Imported: Job ran successfully for that record.
    - IF_ROW_STAT= dup_row_exists: Siebel base table has that record.

    Note: Before they can be used in transactions, all products must be released in Siebel after the EIM import,

    For more information about how to release all products within Siebel, see the Siebel 8113 FP installation instructions (MOS Article ID **880452.1)**.
14. For the products to be present in the S_VOD table, navigate to **Administration - Business Service, Simulator** and execute the business service "ISS Authoring Import Export Service" with method *Post_EIM_Upgrade*.

    Name: ISS Authoring Import Export Service

    Method: Post_EIM_Upgrade

    Name: ReportOnly

    Value: FALSE

For more information about how products can be imported using EIM, see MOS Article ID 485530.1.

14.4.1.3 Moving Data from Siebel Base Tables to Oracle AIA Cross-Reference Tables

You can move data from Siebel base tables to Oracle AIA cross-reference tables using either ODI Designer or a command line prompt.

To move data from Siebel base tables using ODI Designer:

1. Log in to the Designer.
2. Expand the project **Product Initial Load**.
3. Expand the folder **Bulk Load**.
4. Expand the package.
5. Right-click the package titled `Load_Siebel_To_XREF_Data`. Select **Execute** from the menu.
6. Select context as *Product Development* in the **Execution** box.

To move data from Siebel base tables using a command line prompt:

1. Go to the bin folder of ODI HOME in the Command window.
2. Enter the following command:
   - For Windows: Startscen LOAD_SIEBEL_TO_XREF_DATA 001 PRODUCTDEVELOPMENT
   - For UNIX: ./startscen.sh LOAD_SIEBEL_TO_XREF_DATA 001 PRODUCTDEVELOPMENT

To verify the data:

After the process finishes and before moving data from the Siebel EIM table to the Siebel base table, verify that the product data was loaded into the Siebel EIM table by completing these actions:

1. Select xref_column_name, count(*) from xref_data where xref_table_name = 'oramds:/apps/AIAMetaData/xref/ITEM_ITEMID.xref'GROUP BY xref_column_name (to be run in FMW DB).
2. Select count(*) from eim_prod_int (to be run in Siebel DB).
3. Select count(*) from eim_prod_int1 (to be run in Siebel DB).
4. Select count(*) from eim_prodinvloc (to be run in Siebel DB).

   The count should be the same.
5. Verify that the Oracle AIA cross-reference tables are populated with the Oracle EBS and Common ID values for the product.
6. Select count(*) from s_prod_int (to be run in Siebel DB) to verify that the product data was loaded into the Siebel base table after you move data from the Siebel EIM table to the Siebel base table.
7. Select xref_column_name, count(*) from xref_data where xref_table_name = 'oramds:/apps/AIAMetaData/xref/ITEM_ITEMID.xref'GROUP BY xref_column_name (to be run in FMW DB) to verify that the Siebel ID is populated after you move data from the Siebel base table to the Oracle AIA cross-reference table.
8. Verify that the Oracle AIA cross-reference tables are populated with the Oracle EBS, Common, and Siebel ID values for the product.

14.5 Loading Initial Price List Data

Use the bulk load feature to move active price list data, such as the following data, related to order capture from Oracle EBS into Siebel CRM:

- Active header information, such as name, currency, start date, and end date.
- Line information, such as product, list price, start date, and end date.

Bulk loading of price list data is similar to loading other data types in that the bulk load feature extracts price list data from Oracle EBS, transforms it into the EIM interface table data structures, and moves it into the Siebel EIM interface tables. A Siebel loader program moves the data from the EIM interface tables into the application base tables of Siebel CRM.

Bulk loading of price list data is different in that you can also synchronize changes to active price list data in the Oracle EBS database to Siebel CRM database. These changes, following the initial bulk load, can be either creation of new price lists or updates to the header and line information of existing price lists. The incremental bulk loads create or update price lists or the header and line information in the Siebel CRM database, depending on whether the price lists or the header and line information exists in the Siebel CRM database.

For more information about initial data loads, see Chapter 2, "Loading Initial Data."

14.5.1 Loading Price List Bulk Data

The Price List bulk data load consists of three steps:

1. Load Oracle EBS to EIM.

   Use Oracle Data Integrator (ODI) to populate the Siebel EIM table (EIM_PRI_LST) with the data from the Oracle EBS table (QP_CRMINTEG_PRICELIST_V). The Oracle AIA cross-reference table is populated with the Oracle EBS and Common ID values.

2. Transfer EIM data to the Siebel Base table.

   Use an EIM job to transfer the data from EIM to the Siebel Base table.

3. Move values from the Siebel Base table to the Oracle AIA cross-reference table.

   Use ODI to move the Siebel ID values back into the Oracle AIA cross-reference table (XREF_DATA), correlating to the Oracle EBS and Common ID values that were loaded in step 1.

For initial data loads, you must perform some EIM activities.

For more information about the pre- and post-EIM activities and execution, see the *Siebel Enterprise Integration Manager Administration Guide*, Siebel EIM Tables.

14.5.1.1 Moving Data from Oracle EBS to Siebel EIM Tables

You can move data from Oracle EBS to Siebel EIM by running the ODI package either from ODI Designer or a command line prompt.

To move data from Oracle EBS using ODI Designer:

1. Log in to the ODI Designer.
2. Under the project PriceListBulkLoad, run the package LoadEbizPriceListDataToSiebelPkg.

To move data from Oracle EBS using a command line prompt:

1. Go to the bin folder of ODI HOME in the Command window.
2. Run the following ODI scenario:
   - For Windows: startscen.bat LOADEBIZPRICELISTDATATOSIEBEL 001 PRICELISTBULKLOAD
   - For UNIX: ./startscen.sh LOADEBIZPRICELISTDATATOSIEBEL 001 PRICELISTBULKLOAD

To verify the data:

1. After the process finishes, verify that the Price List and Lines data was loaded into the Siebel EIM table by completing these actions:
   a. Select count(*) from QP_CRMINTEG_PRICELIST_V (to be run in EBIZ DB).
   b. Select count(*) from eim_pri_lst (to be run in Siebel DB).
   c. The count should be the same.
2. Verify that the Oracle AIA cross-reference tables are populated with the Oracle EBS and Common ID values for the price list by completing these actions:
   a. Select distinct list_header_id from QP_CRMINTEG_PRICELIST_V (EBIZ DB).
   b. Select count(*) from xref_data, where xref_table_name is something like 'PRICELIST_ID' and xref_column_name is something like 'EBIZ_01'.
   c. Select count(*) from xref_data, where xref_table_name is something like 'PRICELIST_ID' and xref_column_name is something like 'COMMON'.
   d. The count should be the same.

Note: QP_CRMINTEG_PRICELIST_V retrieves records for the pricing organization setup in the profile option QP: Item Validation Organization at the site level. Verify that the profile option is set up correctly.

14.5.1.2 Moving Data from EIM to Siebel Base Tables

To move data from EIM to Siebel base tables:

1. Verify that the Product.ifb, Pricelist.ifb, Customer.ifb, and Asset.ifb files are available in the Admin directory of the installed Siebel server.

   For example, in the D:\ses\siebsrvr\Admin directory
2. For Pricelist.ifb, comment the second process `INCLUDE = "UPDATE INTEGRATION ID"` using a terminate symbol (so that it becomes inactive and only the first process is run), for example:

```
[IMPORT ALL PRICELISTS]
TYPE = SHELL
INCLUDE = "IMPORT PRICELIST"
;INCLUDE = "UPDATE INTEGRATION ID"
```

3. Log in to Siebel E-Business Applications using the Administrator login.

   For example, proper credentials such as SADMIN/SADMIN
4. Go to the **Site Map** icon (the globe that appears at the top left of the home page).

5. Navigate to the Administration - Server Management screen.
6. Click the **Jobs** link and navigate to **Server Management, Jobs**.
7. Click **New** in the Jobs applet.
8. Ensure that the component is set to Enterprise Integration Manager in this new job.
9. Go to **Job Parameters** and enter the following parameter values:
   - Configuration File: Pricelist.ifb
   - Username: The administrator username ex. SADMIN
   - Password: The administrator password ex. SADMIN
   - Error Flag: 1
   - SQL Trace Flag: 8
   - Trace Flag: 1
10. Ensure that the job finishes with a status of *Success*.

    The color of the status bar changes to green.
11. Ensure that the Siebel Business Applications User Interface displays the data properly in the relevant Administration screens, such as Accounts, Contacts, Products, Price List, and so on.
12. To check whether the Siebel EIM loaded all records, check the IF_ROW_STAT column for the corresponding EIM table.

    The IF_ROW_STAT field can have following status values:
    - IF_ROW_STAT = for import: Job to be run or job is running.
    - IF_ROW_STAT = Partially_imported: Some mandatory field is missing for that record.
    - IF_ROW_STAT = Imported: Job ran successfully for that record.
    - IF_ROW_STAT= dup_row_exists: Siebel base table has that record.

14.5.1.3 Moving Data from Siebel Base Tables to AIA Cross-Reference Tables

To create the cross-reference for the Siebel Price List IDs, you can run the ODI package from ODI Designer or a command line prompt.

To create the cross-reference for the Siebel price list IDs from ODI Designer:

1. Log in to the ODI Designer.
2. Under the project PriceListBulkLoad, run the package LoadSiebelPriceListDataToXREFPkg.

To create the cross-reference for the Siebel price list IDs from a command line prompt:

1. Go to the bin folder of ODI HOME in the Command window.
2. Run the following ODI scenario:
   - For Windows: startscen.bat LOADSIEBELPRICELISTDATATOXREF 001 PRICELISTBULKLOAD

- For UNIX: ./startscen.sh LOADSIEBELPRICELISTDATATOXREF 001 PRICELISTBULKLOAD

To verify the data:

After the process finishes, the cross-references for the Siebel Price List IDs are created. Perform these actions to verify the data:

1. Select distinct list_header_id from QP_CRMINTEG_PRICELIST_V (to be run in the Oracle EBS database).
2. Select count(*) from s_pri_lst, where integration_id is not null (to be run in the Siebel database)
3. Select count(*) from xref_data, where xref_table_name is something like 'PRICELIST_ID' and xref_column_name is something like 'SEBL_01'

   The count should be the same.

Note: After running the ODI package to populate the X-REF Database, if required, you can run Siebel EIM again to nullify the integration_id in Siebel. This is an optional step. The format of the ifb file for the second run looks like this:

```
[IMPORT ALL PRICELISTS]
TYPE = SHELL
;INCLUDE = "IMPORT PRICELIST"
INCLUDE = "UPDATE INTEGRATION ID" A
```

All the other job parameters remain the same in the ifb file.

14.5.2 Loading Incremental Price Lists

The purpose of an incremental load is to move new price lists and lines or any updates to existing price lists into Siebel CRM for use in the order-capture process. Use ODI to manage incremental loads.

To perform the incremental load:

1. Use ODI to populate the Siebel EIM table (EIM_PRI_LST) with data from the Oracle EBS table (QP_CRMINTEG_PRICELIST_V) as specified in $AIA_HOME/services/core/BulkDataProcess/EbizToSiebel/PriceList/PriceListSource.xml.

   The Oracle AIA cross-reference table is populated with the Oracle EBS and Common ID values.
2. Run the EIM load process to transfer data to the Siebel Base table.
3. Load the Siebel ID values in the Oracle AIA cross-reference table (XREF_DATA).

14.5.2.1 Populating the Siebel EIM Table

To load Price List data from Oracle EBS as specified in the XML file at $AIA_HOME/staging/O2C2Home/SetupFiles/PriceListSource.xml into Siebel EIM and to create cross-references for Oracle EBS and Common IDs, you can run the following ODI scenario from the directory ODI_HOME/bin using either ODI Designer or a command line prompt.

To run from the ODI Designer:

1. Log in to the ODI Designer.
2. Under the project PriceListBulkLoad, run the package LoadEbizIncPriceListDataToSiebelPkg.

To run from the command line:

- Windows: startscen.bat LOADEBIZINCPRICELISTDATATOSIEBEL 001 PRICELISTBULKLOAD
- LINUX: ./startscen.sh LOADEBIZINCPRICELISTDATATOSIEBEL 001 PRICELISTBULKLOAD

To verify the data:

After the process finishes, the Price List and Lines data is loaded into the Siebel EIM table. Only those price lists specified in the XML file are loaded.:

1. To verify the data, select a unique PL_NAME from eim_pri_lst (Siebel DB). The price list names should match the ones in the XML file.
2. To verify that the Oracle AIA cross-reference tables are populated with the Oracle EBS and Common ID values for the price list, complete this action:

   Select list_header_id from QP_CRMINTEG_PRICELIST_V ($AIA_HOME/staging/O2C2Home/SetupFiles/PriceListSource.xml) (to be run in the Oracle EBS database) and ensure that the IDs returned are in the cross-reference table.

14.5.2.2 Running the Siebel EIM Load

To run the Siebel EIM load:

1. Verify that the Product.ifb, Pricelist.ifb, Customer.ifb, and Asset.ifb files are available in the Admin directory of the installed Siebel server.

   For example, in the D:\ses\siebsrvr\Admin directory

2. For Pricelist.ifb, comment the second process using a terminate symbol (so that it becomes inactive and only the first process is run). For example:

```
[IMPORT ALL PRICELISTS]
TYPE = SHELL
INCLUDE = "IMPORT PRICELIST"
;INCLUDE = "UPDATE INTEGRATION ID"
```

3. Log in to the Siebel E-Business Application using the Administrator login, for example, proper credentials such as SADMIN/SADMIN
4. Go to the **Site Map** icon (the globe that appears at the top left of the home page).
5. Navigate to the Administration - Server Management screen.
6. Click the **Jobs** link and navigate to **Server Management, Jobs**.
7. Click **New** in the Jobs applet.
8. Ensure that the component is set to Enterprise Integration Manager in this new job.
9. Go to the **Job Parameters** applet and enter the following parameter values:
   - Configuration File: The exact name of the ifb file, for example, Product.ifb, Pricelist.ifb and so on.

- Username: The administrator username, for example, SADMIN
- Password: The administrator password, for example, SADMIN
- Error Flag: 1
- SQL Trace Flag: 8
- Trace Flag: 1

10. Ensure that the job finishes with a status of *Success*.
11. Ensure that the data appears correctly in the Siebel Business Applications User Interface in relevant Administration screens, for example, Accounts, Contacts, Products, Price List, and so on.
12. To check whether the Siebel EIM loaded all records, check the IF_ROW_STAT column for the corresponding EIM table.

    The IF_ROW_STAT field can have following status values:

    - IF_ROW_STAT = for import: Job to be run or job is running.
    - IF_ROW_STAT = Partially_imported: A mandatory field is missing for that record.
    - IF_ROW_STAT = Imported: Job ran successfully for that record.
    - IF_ROW_STAT= dup_row_exists: Siebel base table has that record.

14.5.2.3 Creating Siebel Cross-References

To create the cross-references for the Siebel price list IDs for those price lists specified in $AIA_HOME/staging/O2C2Home/SetupFiles/PriceListSource.xml, you can run the ODI scenario from ODI_HOME/bin either from ODI Designer or a command line prompt.

To run from the ODI Designer:

1. Log in to ODI Designer.
2. Under the project PriceListBulkLoad, run the package LoadSiebelIncPriceListDataToXREFPkg.

To run from the command line:

- For Windows:startscen.bat LOADSIEBELINCPRICELISTDATATOXREF 001 PRICELISTBULKLOAD
- For LINUX: ./startscen.sh LOADSIEBELINCPRICELISTDATATOXREF 001 PRICELISTBULKLOAD

To verify data after load:

1. After the process finishes, verify that the cross-references for the Siebel Price List IDs were created by selecting row_id from s_pri_lst, where the pricelist names are in the xml file ($AIA_HOME/staging/O2C2Home/SetupFiles/PriceListSource.xml) (to be run in Siebel database).
2. Ensure that the returned IDs are created in the Oracle AIA cross-reference table.

Note: After running the ODI package to populate the X-REF Database, if needed, you can run Siebel EIM again to nullify the integration_id in Siebel. This step is optional. The format of the IFB file for the second run looks like this:

```
[IMPORT ALL PRICELISTS]
TYPE = SHELL
;INCLUDE = "IMPORT PRICELIST"
INCLUDE = "UPDATE INTEGRATION ID"
```

All the other job parameters remain the same in the ifb file.

14.5.3 Loading Price List Data from Oracle EBS to Siebel EIM Overview

Pricing administrators are responsible for:

1. Specifying the list of price list names in the XML file at $AIA_HOME/staging/O2C2Home/SetupFiles/PriceListSource.xml for the incremental load.
2. Populating cross-references with Oracle Price List ID and Common IDs for the price lists in the XML file at $AIA_HOME/staging/O2C2Home/SetupFiles/PriceListSource.xml by running an ODI package.
3. Loading the specified price lists, new or updated, into the Siebel interface tables (EIM_PRI_LST) by running an ODI package.
4. Running the Siebel EBO implementation map (EIM) program to populate the Siebel base tables.
5. Linking cross-references with the Siebel ID by running an ODI package.

For more information about initial data loads, see Chapter 2, "Loading Initial Data."

14.5.3.1 Loading Price List Data from Oracle EBS to Siebel EIM

Source Schema Description

- Source Application: Oracle E-Business Suite
  - QP_CRMINTEG_PRICELIST_V

Target Schema Description

- Target Application: Siebel
  - Table: EIM_PRI_LST

ODI Mapping Design

QP_CRMINTEG_PRICELIST_V is the view object to fetch all sales order-related active price list headers and lines.

1. Populate the cross-reference with the Price List ID and Common ID:
   - ODI package: LoadEbizIncPriceListDataToSiebelPkg
   - ODI Scenario: LOADEBIZINCPRICELISTDATATOSIEBEL
   - ODI Interface: Ebiz_To_Xref_PriceList_INCINTERFACE
   - Source: CRMINTEG_PRICELIST_V

- Target: cross-reference table
- Join condition: Select * from QP_CRMINTEG_PRICELIST_V

Table 14–2 ODI Mapping

Column in Target	Column in Source (Table)
SEBL_01	<< not mapped>>
ORCL_01	LIST_HEADER_ID
COMMON	Unique sequence number

2. Load the EIM table:
   - ODI package: LoadEbizIncPriceListDataToSiebelPkg
   - ODI Scenario: LOADEBIZINCPRICELISTDATATOSIEBEL
   - ODI Interface: Ebiz_To_Sebl_PriceList_INCINTERFACE
   - Source: QP_CRMINTEG_PRICELIST_V
   - Join condition: Select * from QP_CRMINTEG_PRICELIST_V WHERE (name in price list names mentioned in the xml)
   - Target: EIM_PRI_LST

Siebel EIM Configuration File

See Example 14–1, "Siebel EIM Configuration File".

Example 14–1 Siebel EIM Configuration File

```
[Siebel Interface Manager]
USER NAME = "SADMIN"
PASSWORD = "SADMIN"
PROCESS = "IMPORT ALL PRICELISTS"
[IMPORT ALL PRICELISTS]
TYPE = SHELL
INCLUDE = "IMPORT PRICELIST"
INCLUDE = "UPDATE INTEGRATION ID"
[IMPORT PRICELIST]
BATCH = 100
TYPE = IMPORT
TABLE = EIM_PRI_LST
ONLY BASE TABLES = S_PRI_LST,S_PRI_LST_ITEM,S_PRI_LST_BU
FIXED COLUMN = PL_ENTERPRISE_FLG,"N"
FIXED COLUMN = PLI_OVERRIDEROLLUP,"N"
FIXED COLUMN = PLI_PRI_CD,"STANDARD"
FIXED COLUMN = VIS_ACTIVE_FLG,"Y"
FIXED COLUMN = PL_ACTIVE_FLG,"Y"
DEFAULT COLUMN = PL_BU,"Vision Operations"
DEFAULT COLUMN = VIS_BU,"Vision Operations"
DEFAULT COLUMN = PLI_PROD_BU,"Vision Operations"
DEFAULT COLUMN = PL_EFF_START_DT,"2007-01-15"
DEFAULT COLUMN = PLI_EFF_START_DT,"2007-01-15"
DEFAULT COLUMN = PL_TAXABLE_FLG,"N"
DEFAULT COLUMN = PL_SUBTYPE_CD,"PRICE LIST"
[UPDATE INTEGRATION ID]
SESSION SQL = "UPDATE ORA19111.EIM_PRI_LST SET PL_INTEGRATION_ID = NULL WHERE PL_
INTEGRATION_ID IS NOT NULL"
BATCH = 100
```

```
TYPE = IMPORT
TABLE = EIM_PRI_LST
ONLY BASE TABLES = S_PRI_LST
ONLY BASE COLUMNS = S_PRI_LST.NAME,\
S_PRI_LST.BU_ID,\
S_PRI_LST.SUBTYPE_CD,\
S_PRI_LST.INTEGRATION_ID
INSERT ROWS = S_PRI_LST,FALSE
UPDATE ROWS = S_PRI_LST,TRUE
NET CHANGE = FALSE
```

14.5.3.2 Loading the Cross-Reference Table

After Siebel EIM tables are populated, you run the Siebel server EIM component to populate the Siebel base tables. After the price lists are created in Siebel, you must update the Siebel Price List ID and the Price List Line ID in the X-REF table.

For more information about initial data loads, see Chapter 2, "Loading Initial Data."

Source Schema Description

- S_PRI LST

Target Schema Description

- X-REF table

ODI Mapping Design

- The S_PRICE LIST table has ROW_ID and INTEGRATION_ID that must be mapped to the cross-reference table.
  - ODI Package name: LoadSiebelIncPriceListDataToXREFPkg
  - ODI Scenario Name: LOADSIEBELINCPRICELISTDATATOXREF
  - ODI interface name: Sebl_To_Xref_PriceList_INCINTERFACE
  - Source: S_PRI_LST
  - Target: X-REF Table

Table 14–3 ODI Mapping

Column in Target	Column in Source (Table)
SEBL_01	ROW_ID (S_PRI LST)
ORCL_01	Should be populated during the Oracle view to EIM map.
COMMON	INTEGRATION_ID (S_PRI_LST)

14.6 Loading Initial Assets Data

Use the bulk load feature to move asset data (related to customer-owned item instances) from Oracle EBS into a Siebel CRM Asset.

Bulk loading of asset data is similar to that of other data types, such as customer and product. The bulk load feature extracts asset data from Oracle EBS, transforms it into the EIM interface table data structures, and moves it into the Siebel EIM interface tables. A Siebel loader program moves the data from the EIM interface tables into the application base tables of Siebel CRM.

14.6.1 Populating Initial Data for Cross-Reference

The INVENTORY_LOCATION_ID cross-reference table must be populated manually after the installation because no process flow exists for this synchronization.

shows how to use the SQL insert statement to manually add the data into the INVENTORY_LOCATION_ID cross-reference table.

Example 14–2 Adding Data Into the INVENTORY_LOCATION_ID XREF Table

```
SEBL01_INVLOC_BU:
INSERT INTO XREF_DATA (XREF_TABLE_NAME,XREF_COLUMN_NAME,
ROW_NUMBER, VALUE, IS_DELETED, LAST_MODIFIED) VALUES
('oramds:/apps/AIAMetaData/xref/INVENTORY_LOCATION_ID.xref','SEBL01_INVLOC_BU'
,
'ROWNUM_ORG_1', '88-JKO29', 'N', SYSTIMESTAMP)
```

Note: The SEBL01_INVLOC_BU column must be populated with the Business Unit Id value from Siebel.

14.6.2 Loading Asset Bulk Data

The Asset bulk data load consists of these steps:

1. Load Oracle EBS to EIM.

   Use ODI to populate the Siebel EIM table with data from the Oracle EBS table. The Oracle AIA cross-reference table is populated with the Oracle EBS and Common ID values.

2. Transfer EIM data to a Siebel Base table.

   Use an EIM job to transfer the data from EIM to the Siebel Base tables.

3. Move Siebel Base table values to the Oracle AIA cross-reference table.

   Use ODI to move the Siebel ID values back into the Oracle AIA cross-reference table (XREF_DATA), correlating to the Oracle EBS and Common ID values that were loaded in step 1.

For initial data loads, you must perform some EIM activities.

For more information about the pre- and post-EIM activities and execution, see the *Siebel Enterprise Integration Manager Administration Guide*, Siebel EIM Tables.

14.6.2.1 Moving Data from Oracle EBS to Siebel EIM Tables

You can move data from Oracle EBS to Siebel EIM tables using either ODI Designer or a command line prompt.

To move data from Oracle EBS using ODI Designer:

1. Log in to the Designer.
2. Expand the project: **Oracle Ebiz to Siebel Project**.
3. Expand the folder: **Assets Bulk Load**.
4. Expand the package.
5. Right-click the package Load Ebiz Asset Data to Eim Pkg and select **Execute** from the menu.
6. Select *Global* in the execution box.

To move data from Oracle EBS using a command line prompt (Oracle EBS 12.1.1):

1. Go to the bin folder of ODI HOME in the Command window.
2. Enter the following command:
   - For WINDOWS: startscen LOAD_EBIZ_ASSET_DATA_PKG_R12 001 GLOBAL
   - For UNIX: ./startscen.sh LOAD_EBIZ_ASSET_DATA_PKG_R12 001 GLOBAL

To move data from Oracle EBS using a command line prompt (Oracle EBS 11.5.10):

1. Go to the bin folder of ODI HOME in the Command window.
2. Enter the following command:
   - For WINDOWS: startscen LOAD_EBIZ_ASSET_DATA_TO_EIM_PKG 001 GLOBAL
   - For UNIX: ./startscen.sh LOAD_EBIZ_ASSET_DATA_TO_EIM_PKG 001 GLOBAL

14.6.2.2 Moving Data from EIM to Siebel Base Tables

To move data from EIM to Siebel base tables:

1. Verify that the Product.ifb, Pricelist.ifb, Customer.ifb, and Asset.ifb files are available in the Admin directory of the installed Siebel server.

   For example, in the D:\ses\siebsrvr\Admin directory
2. Log in to Siebel E-Business Applications using the Administrator login.

   For example, proper credentials such as SADMIN/SADMIN
3. Go to the **Site Map** icon (the globe that appears at the top left of the home page).
4. Navigate to the Administration - Server Management screen.
5. Click the **Jobs** link and navigate to **Server Management, Jobs**.
6. Click **New** in the Jobs applet.
7. Ensure that the component is set to Enterprise Integration Manager in this new job.
8. Go to **Job Parameters** and enter the following parameter values:
   - Configuration File: Assets.ifb
   - Username: The administrator username ex. SADMIN
   - Password: The administrator password ex. SADMIN
   - Error Flag: 1
   - SQL Trace Flag: 8
   - Trace Flag: 1
9. Ensure that the job finishes with a status of *Success*.

   The color of the status bar changes to green.

10. Ensure that the data is displayed properly in the Siebel Business Applications User Interface in relevant Administration screens such as Accounts, Contacts, Products, Price List, and so forth.
11. To check whether the Siebel EIM loaded all records, check the IF_ROW_STAT column for the corresponding EIM table.

    The IF_ROW_STAT field can have the following status values:

    - IF_ROW_STAT = for import: Job to be run or job is running.
    - IF_ROW_STAT = Partially_imported: A mandatory field is missing for that record.
    - IF_ROW_STAT = Imported: Job ran successfully for that record.
    - IF_ROW_STAT= dup_row_exists: Siebel base table has that record.

14.6.2.3 Moving Data from Siebel Base Tables to AIA Cross-Reference Tables

You can move data from Siebel base tables to AIA cross-reference tables using either ODI Designer or a command line prompt.

To move data from Siebel base tables using ODI Designer:

1. Log in to the Designer.
2. Expand the project: **Oracle Ebiz to Siebel Project**.
3. Expand the folder: **Assets Bulk Load**.
4. Expand the package.
5. Right-click the package titled `Load Siebel Data to XREF Pkg`. Select **Execute** from the menu.
6. Select context as the global in the execution box.

To move data from Siebel base tables using a command line prompt:

1. Go to the bin folder of ODI HOME in the Command window.
2. Enter the following command:
    - For Windows: Startscen LOAD_SIEBEL_DATA_TO_XREF_PKG 001 GLOBAL
    - For UNIX: ./startscen.sh LOAD_SIEBEL_DATA_TO_XREF_PKG 001 GLOBAL

To verify the data:

1. After the process finishes and before moving data from the Siebel EIM table to the Siebel base table, run this sql query in the xref database to count the number of records loaded upon running the first ODI scenario of Asset Initial load:
2. Select count(*),Xref_column_name from xref_data

    Where xref_table_name='INSTALLEDPRODUCT_ID' group by xref_column_ name. The count of 'COMMON' and 'EBIZ_01' should be the same.
3. Run this query in the Siebel database (after running the second ODI scenario) to count the number of records in the EIM table of the Siebel:
4. Select count(*),Xref_column_name from xref_data

    Where xref_table_name='INSTALLEDPRODUCT_ID' group by xref_column_ name. This option gives the count of E-Business records loaded into Siebel, along with their Common IDs count. The count should match.

5. Select xref_column_name, count(*) from xref_data, where xref_table_name = 'INSTALLED_PRODUCT_ID' GROUP BY xref_column_name (to be run in FMW database).
6. If the load failed before the data is loaded into the Siebel base tables (through EIM jobs), you can go directly and delete the data from the corresponding EIM tables and then delete the data from the cross-reference for that load and rerun the load.
7. If the process fails after you run the EIM job, you must clear the data from the Siebel database tables and then continue with step 1.

The IF_ROW_STAT column can have these statuses: failed, 'for import', duplicate, partially imported

Initially the data is loaded into the EIM tables with IF_ROW_STAT set as 'for import' when you run the EIM job. Then, based on whether that record is loaded into the Siebel database perfectly, the status changes. Duplicate status means that the data is present in the Siebel database so that the record is not loaded into the Siebel base tables. Failed status means that the import of the record was unsuccessful. Partially failed status means that there is some data inconsistency in the EIM table.

15

Configuring the Order to Cash Pre-Built Integration

This chapter provides instructions about how to configure the Order to Cash: Siebel CRM - EBS pre-built integration. This includes configuring Siebel Customer Relationship Management (Siebel CRM) and Oracle E-Business Suite (Oracle EBS) as well as setting up organizations, inventory locations, Oracle Configurator, and so on.

This chapter includes the following sections:

15.1 Configuring Siebel CRM

After installing and configuring Siebel CRM and any required patches, perform the following configuration steps within your Siebel CRM system.

For more information about required patches, see *Oracle Application Integration Architecture Installation and Upgrade Guide for Pre-Built Integrations*, "Software Requirements."

To configure Siebel CRM for the Order to Cash: Siebel CRM - EBS integration:

Follow these steps to configure the Oracle Fusion Middleware (FMW) URLs for the outbound web services in the Siebel CRM application.

1. Go to **Site Map**.
2. Click **Administration - Web Services**.
3. Click **Outbound Web Service**.
4. In the **Name** field of the web services applet, query for **JMS* or *ABCS* or *ESB* or *ConfiguratorUserLangSiebelAdapter**. The results list 14 web services.

   Of the results, consider the following services:

   - SyncCustomerSiebelEventAggregator
   - RequestProductStructureSiebelJMSProducer
   - ProcessSalesOrderSiebelJMSProducerV2 (for Siebel 8.0.0.x only)
   - CreditCheckSalesOrderSiebelReqABCSImplService
   - CalculateShippingChargeSalesOrderSiebelReqABCSImplService
   - PaymentAuthorizationSalesOrderSiebelReqABCSImplService
   - ProcessQuoteSiebelJMSProducer (for Siebel 8.0.0.x only)
   - CheckATPSalesOrderSiebelReqABCSImplService
   - ConfiguratorUserLangSiebelAdapter
   - ESB_ConfiguratorCopyConfigEbizAdapter_Service
5. Get the FMW URLs for these services. Update the Address column of the Service Port applet to point to the correct FMW server and port number. For example:

   ```
   http://<host name>:<port number>/soa-infra/services/default/
   CheckATPSalesOrderSiebelReqABCSImpl/CheckATPSalesOrderSiebelReqABCSImpl
   <table name="ORGANIZATION_ID">
   ```

6. Activate the workflows listed in Table 15–1.

Table 15–1 Workflow Listing

Workflow Name	Status	Project
CZ PSP Interface	New	Web Service Integration
Account - New Order	Modified	COM Workflows
CZ Return	New	Web Service Integration
Oracle Configurator Load	New	Web Service Integration
Oracle Verify Complex Product All (Order)	New	Web Service Integration
Oracle Verify Complex Product All (Quote)	New	Web Service Integration
Oracle Verify Header (Order)	New	Web Service Integration
Oracle Verify Header (Quote)	New	Web Service Integration
Oracle Verify Item (Order)	New	Web Service Integration
Oracle Verify Item (Quote)	New	Web Service Integration

Table 15–1 (Cont.) Workflow Listing

Workflow Name	Status	Project
SWI Account Update Workflow	New	Web Service Integration
SWI Address Update Workflow	New	Web Service Integration
SWI Contact Update Workflow	New	Web Service Integration
SWI External Product Sync Workflow	New	Web Service Integration
SWIOrderUpsert	New	Web Service Integration
SWIQuoteUpsert	New	Web Service Integration
SWISendATPCheck	New	Web Service Integration
SWISendATPCheckLine	New	Web Service Integration
SWISendCalculateShippingCharge	New	Web Service Integration
SWISendCreditCheck	New	Web Service Integration
SWISendOrder	New	Web Service Integration
SWISendPaymentAuthorization	New	Web Service Integration
SWISendQuote	New	Web Service Integration
Submit Order ASI	New	COM Workflows
Submit Quote ASI	New	COM Workflows

For more information about activating workflows, see *Siebel Business Process Framework: Workflow Guide*, available on Oracle Technology Network.

7. Enable the component groups listed in Table 15–2.

Table 15–2 Component Groups Listing

Component Group Name	Alias
Enterprise Application Integration	EAI
Siebel High Tech Industrial Manufacturing	HTIM
Siebel ISS	ISS
Workflow Management	Workflow

Siebel CME (Communications) is also supported.

For more information about enabling component groups, see *Siebel System Administration Guide*, available on Oracle Technology Network.

15.2 Setting Up Organizations and Inventory Locations

In Oracle EBS, an operating unit is a logical organization within a company that the company management decides to operate. Order transactions are owned by the operating unit organization.

The transactions for an operating unit are restricted to using the reference data for that same operating unit. That is, all the sales orders (transactional entity) are not only

owned by the operating unit on the transaction side, but the reference data is also owned (namely customer accounts or associated items).

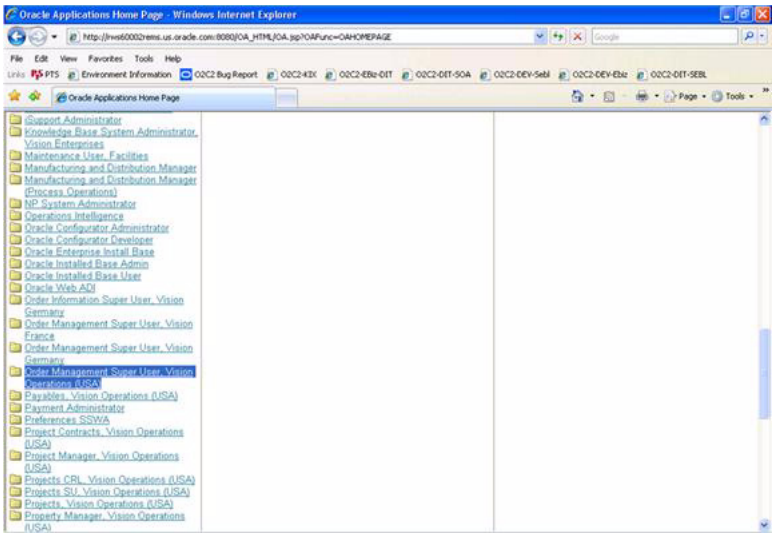
Inventory Organizations in Oracle EBS represent manufacturing and storage facilities. Each inventory organization belongs to one parent operating unit. Oracle implements storage facilities, warehouses, and distribution centers in inventory organizations.

15.2.1 Getting Inventory Location Details in Oracle EBS

To get inventory location details in Oracle EBS:

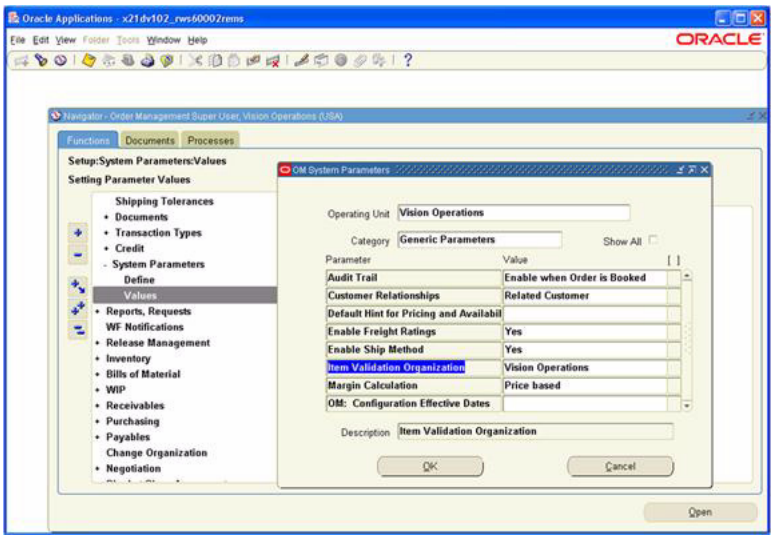
1. Log in to Oracle EBS.
2. Confirm that you have the Order Management (OM) responsibilities for the operating units of interest, as shown in Figure 15–1.

Figure 15–1 Example of the Oracle Applications Home Page



3. Perform the following steps for each of the OM responsibilities that are of interest to you:
   a. Select the **System Parameters** menu item.
   b. Query the system parameters, as shown in Figure 15–2.
   c. Note the name of the item validation organization associated with the parameter item validation organization, for example, Vision Operations, Vision Germany, and Vision France.

Figure 15–2 Example of System Parameters Query



15.2.2 Setting Up Organizations in Siebel CRM

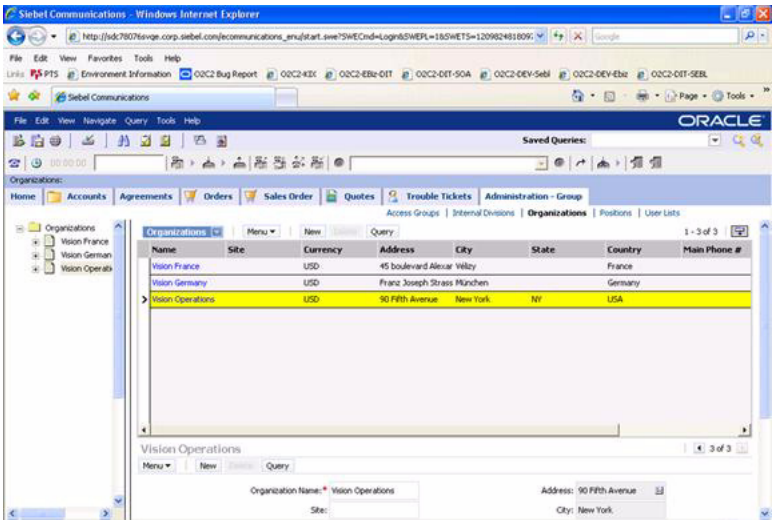
The **Business Unit** organization in Siebel CRM allows the implementation company to partition itself into logical groups. The information is then associated to the business unit, and can be viewed by only the end-users who are associated to that business unit (BU).

The transaction data in Siebel CRM (namely Sales Order) is always associated to a BU (the primary BU). In Siebel CRM, although an order is associated to a specific BU, products from different BUs can be associated on the order lines. In other words, unlike Oracle EBS, the reference data for a transaction can belong to a different organization in Siebel CRM.

To map Siebel organizations to Oracle EBS operating units:

1. Log in to Siebel Applications.
2. Click **Site Map**.
3. Select **Administration - Group, Organizations**.
4. For the Oracle EBS operating units that you identified previously, create the same in Siebel CRM, as shown in Figure 15–3.

Figure 15–3 Example of Setting Up Organizations in Siebel CRM



Note: Create an organization record with name only. Leave the **Site** field blank.

15.2.3 Setting Up Inventory Locations in Siebel CRM

In Siebel CRM, **Inventory Locations** identify where products are stored and the source from which the product is fulfilled. An inventory location may be a warehouse, a field office, or it may be virtual location. An inventory location is also associated to a BU.

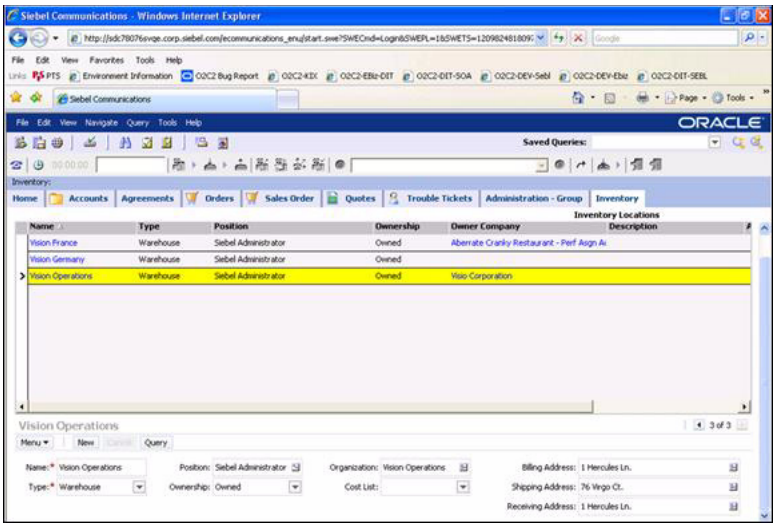
To set up inventory locations in Siebel CRM:

1. Log in into Siebel Applications.
2. Click **Site Map**.
3. Select Inventory / All Inventory Locations across Organizations.

Note: In Oracle, the Ids and names are shared among operating units and inventory organizations.

4. Create the inventory location using the same name that was identified in Oracle EBS, as shown in Figure 15–4.

Figure 15–4 Example of Inventory Locations



5. Ensure that the organization and inventory location name in Siebel CRM matches the names in Oracle EBS because the bulk load references them by name and not by Id.

15.3 Setting Up Cross-References for Siebel IDs and Oracle EBS Entities

After creating inventory locations in both Siebel CRM and in Oracle EBS, create cross-references.

> **Note:** In Siebel CRM, you create inventory locations manually.

15.3.1 Identifying Siebel Row IDs

To set up a cross-reference:

1. Log in to the Siebel database as the table owner.
2. Run the following query to get the Ids for the organizations created in the previous step:

```
select row_id, name from s_org_ext where name like '%Vision%'
```

3. Run the following query to get the Ids for the inventory locations created previously.

```
select row_id, name from s_invloc where name like '%Vision%')
```

15.3.2 Identifying EBS Entities

To get the operating unit details:

1. Log in to the Oracle EBS database (Apps/Apps).
2. Identify the operating units that must be synchronized or maintained in Oracle EBS.
3. Log in to Oracle Applications and get the name for the operating units.

For example: Vision Operations (204), Vision Germany (888), Vision France (911)

4. To pick other operating units, use the following query:

```
select organization_id, name from hr_operating_units
```

15.3.3 Populating Initial Data for Cross-References

To populate the initial date for cross-references

You must manually populate the ORGANIZATION_ID and INVENTORY_LOCATION_ID cross-reference table after the install because no process flow exists for this synchronization. Manually add the data into this table using the sql insert statements. The ORGANIZATION_ID table holds the information for business units among the edge applications and the common value.

Sample sql statement (for ORGANIZATION_ID table):

- EBIZ_01

  INSERT INTO XREF_DATA (XREF_TABLE_NAME,XREF_COLUMN_NAME, ROW_NUMBER, VALUE, IS_DELETED, LAST_MODIFIED) VALUES ('oramds:/apps/AIAMetaData/xref/ORGANIZATION_ID.xref','EBIZ_01', 'ROWNUM_ORG_1', '204', 'N', SYSTIMESTAMP)

- SEBL_01

  INSERT INTO XREF_DATA (XREF_TABLE_NAME,XREF_COLUMN_NAME, ROW_NUMBER, VALUE, IS_DELETED, LAST_MODIFIED) VALUES ('oramds:/apps/AIAMetaData/xref/ORGANIZATION_ID.xref','SEBL_01', 'ROWNUM_ORG_1', '88-JKO29', 'N', SYSTIMESTAMP)

- COMMON

  INSERT INTO XREF_DATA (XREF_TABLE_NAME,XREF_COLUMN_NAME, ROW_NUMBER, VALUE, IS_DELETED, LAST_MODIFIED) VALUES ('oramds:/apps/AIAMetaData/xref/ORGANIZATION_ID.xref','COMMON', 'ROWNUM_ORG_1', '1001001', 'N', SYSTIMESTAMP)

For each set, which includes EBIZ_01, SEBL_01, and COMMON, the row numbers must be identical to link these records to each other.

For more information on populating cross-references, see the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*, "Working with Cross References".

15.3.4 Validating Cross-References

To validate the cross-references:

1. Log in to the Oracle Application Integration Architecture (Oracle AIA) XREF database.
2. Query the table XREF_DATA (as shown in Example 15–1) to confirm that every organization and every inventory location used in the XML files has three records.

Example 15–1 Querying Table XREF_DATA

```
select value||'::'||Xref_column_name from xref_Data where
row_number in (select row_number from xref_data where
xref_table_name = 'oramds:/apps/AIAMetaData/xref/INVENTORY_LOCATION_ID.xref'
and value in ('204'))
```

3. Replace the value for the inventory locations and replace INVENTORY_LOCATION_ID with ORGANIZATION_ID for the operating units that you selected. (The number of operating units and inventory locations depends on your set up.)

For more information about cross-references, see Section 15.7, "Working with Cross-References."

15.4 Setting Up Additional Business Event Subscriptions

In Oracle EBS, items are created in the master organization and assigned to inventory organizations from the item master organization. In the delivered integration, these default subscriptions are created for two events raised for item create and update:

- oracle.apps.ego.item.postItemUpdate
- oracle.apps.ego.item.postItemCreate

These subscriptions trigger the item synchronization flow for every item created or updated in any inventory organization. The integration flow itself only propagates the item created or updated to Siebel CRM if the inventory organization is mapped in the INVENTORY_LOCATIONS X-Ref table.

It is highly recommended to optimize the default subscriptions to only trigger the item create and update flows for the subset of inventory organizations relevant for order processing in the integration. Typically, this is the set of item validation organizations.

For more information about considerations for organization set up and use case scenarios, see Appendix B, "Organization Data Setup for Product Synchronization."

By default, the subscription for the two events mentioned previously, are processed by the PLSQL function: "aia_custom_subscription_pkg.aia_item_subscription". This package is created in Oracle EBS when the integration is deployed.

In addition, another PLSQL package titled aia_item_subscrpt_params_pkg specifies parameters used by the above subscription PLSQL package. The default source for this package is available in the $AIA_HOME/services/core/Ebiz/AdapterServices/CreateItemEbizEventConsumer/aia_item_subscription_params_pkg.sql file.

The following are the three variables in the aia_item_subscrpt_params_pkg package that must be modified.

- g_validate_subscription
- g_master_organization_id
- g_itemValidationOrgs

To set up subscriptions:

1. Modify the three variables:

   a. Set the g_validate_subscription value to either *Y* or *N*.

      N (default): This value implements the as delivered behavior. For example, the subscription triggers the item synchronization flow for every item created or updated in any inventory organization.

      Y: The subscription triggers the item synchronization flow for only items that were created or updated in the master or item validation organizations specified in the other two variables.

b. Set the g_master_organization_id value to the Id of the item master organization. For example: g_master_organization_id NUMBER:=204;

c. Set the g_itemValidationOrgs value to the list of item validation org Ids. For example: g_itemValidationOrgs NumberList:=NumberList(204,911);

If you use the examples shown here, item creates and updates are synchronized for 204 and 911 organizations.

2. Recompile the aia_item_subscript_params_pkg package against the Oracle EBS database (Applications schema).

15.5 Setting Up Application Context Definitions for Oracle EBS

This section describes how to set up the derivation of the context in flows in which Oracle EBS application programming interface (API) or service calls are made.

Application context (or API) for Oracle EBS calls are made up of three components:

- Operating Unit
- Username
- Responsibility

Operating units should be mapped to the corresponding entities in other applications (such as business units in Siebel CRM), as described in Section 15.3, "Setting Up Cross-References for Siebel IDs and Oracle EBS Entities" and cross-references in the ORGANIZATION_ID XREF table. In any flow in which the organization context of the source application is available in the flow and is mapped in the ORGANIZATION_ID XREF table, looking up the ORGANIZATION_ID XREF table derives the operating unit. If any context is not mapped or cannot be looked up, then a default is used. This default is set up in the following configuration property:

Property Name: TransformAppContextEbizService.DefaultOperatingUnit

Service Name: TransformAppcontextEbizService

If required, users can also create cross-references between applications using the USER_NAME XREF table. In any flow in which the user context of the source application is available and is mapped in the USER_NAME XREF table, looking up the USER_NAME XREF table derives the username for Oracle EBS. If any context is not mapped or cannot be looked up, then a default is used. This default is set up in the following configuration property:

Property Name: TransformAppContextEbizService.DefaultUser

Service Name: TransformAppcontextEbizService

Responsibilities should be mapped to the Oracle EBS user + operating unit combination. The responsibility should be valid with privileges to operate in the operating unit, and must be assigned to this user. This mapping is done in the ORACLE_RESPONSIBILITY DVM. The column EBIZ_USER_OU should have the concatenation of USER_NAME:ORGANIZATION_ID and the column EBIZ_RESP should have the corresponding responsibility to be used.

For example,

EBIZ_USER_OU - OPERATIONS: 204
EBIZ_RESP - Order Management Super User, Vision Operations (USA)

15.6 Setting Up Oracle Configurator

For the Quotes and the Orders integrations, you must change the run-time configuration in Siebel CRM to invoke the Oracle Configurator from Siebel CRM.

For more information about how to set up Siebel QF for ACR 508 see the instructions for Siebel 8.1.1.3 FP (MOS Article ID **880452.1)**.

15.6.1 Changing the Run-Time Configuration to Invoke Oracle Configurator from Siebel CRM

You must make the following run-time configuration changes before you can launch Oracle Configurator.

To change the run-time configuration to invoke Oracle Configurator from Siebel CRM:

1. Log in to the Siebel HTIM application from a browser.
2. Go to **Site Map, Administration Business process, Repository workflow process**.
   a. In the top applet, query for the workflows imported previously and activate them.
   b. Ensure that they are in the active state by querying in the bottom applet.
3. Go to S**ite Map, Administration Order Management, Signals**.
   a. Query for the signal *Customize*.
   b. Lock the signal record and click **Workspace**.
   c. Change the service name to *Oracle Configurator Load*.
   d. Navigate back to the Signals view and click the **Release New Version** button to release a new version of the signal.
   e. Query for the signal *QuotesAndOrdersValidate*.
   f. Lock the signal record and click **Workspace**.
   g. Replace the Siebel Verification with Oracle Batch Validate. Change all four records with sequence #2:

      Mode = Quote, Instance Type = Line Item - Change Service Name column to "Oracle Verify Item (Quote)"

      Mode = Order, Instance Type = Line Item - Change Service Name column to "Oracle Verify Item (Order)"

      Mode = Quote, Instance Type = Header - Change Service Name column to "Oracle Verify Header (Quote)"

      Mode = Order, Instance Type = Header - Change Service Name column to "Oracle Verify Header (Quote)"
4. Go to **Site Map, Administration Runtime Events, Events**, and click **Menu, Reload Runtime Events**.
5. Go to **Site Map, Administration - Application, Views**.
   a. Click **New** to add the following new views:

      CZRuntimeInstanceView (JS)

      CZRuntimeInstanceView (JS) - Agreement

CZRuntimeInstanceView (JS) - Order (Sales)

CZRuntimeInstanceView (JS) - Favorites

b. For each view, assign the appropriate responsibility so that users with that responsibility can navigate to this view.

6. Go to **Site Map, Administration - Integration, WI Symbolic URL List**.

a. From the top applet, select *Host Administration* from the available options menu.

b. Add a new host entry:

Host Name: <Oracle EBIZ host name:port number>

Virtual Name: OracleConfigurator

c. Select **Symbolic URL Administration** and add a new record in the top applet:

Name: OracleCfgURL

Note: This symbolic name is important because the server-side business component code relies on this name.

URL: http://OracleConfigurator/OA_HTML/CfgSebl.jsp

Select the host name.

Fixup Name: Default

SSO Disposition: IFRAME

d. Add the following arguments to the URL in the bottom applet:

Create a new record with these values:
Name: InitMessage
Req Arg: Selected
Argument Type: Profile Attribute
Argument Value: CZInitMessage
Append as argument: Selected

e. Add another new row and enter the following information:

Name: PostRequest
Required Arg: Selected
Argument Type: Command
Argument Value: PostRequest
Append as argument: Deselected

f. Implement steps g and h to set up the single sign-on (SSO) login to Oracle EBS.

The SSO login is the preferred mode of accessing Oracle EBS. If an SSO login is not set up, then the Siebel end user must enter Oracle login credentials at a login page. If SSO is going to be used, you must complete step 5.

g. For the next two arguments, enter the specified values for Argument Value.

UserLoginId and UserLoginPassword are function names used by Siebel SSO. These functions look up the Oracle login credentials for the current Siebel user. Entering any other value requires the user to log in to an Oracle Applications

for Configurator session. In step 5, map the Siebel user to the Oracle login credentials.

h. Continue to add arguments to the same URL:

Name: Username
Required Arg: Selected
Argument Type: Command
Argument Value: UserLoginId
Append as argument: Selected

i. Add the final argument

Name: Password
Required Arg: Selected
Argument Type: Command
Argument Value: UserLoginPassword
Append as argument: Selected

j. Select **SSO Systems Admin List** and add a record in the top applet:

System Name: OracleConfigSSO
Symbolic URL Name: OracleCfgURL
Description: Logs in to the Oracle Configurator

k. In the bottom applet, SSO System Users, add records for the Siebel users who invoke Oracle Configurator.

Pair the Siebel usernames with Oracle login credentials, for example:

Siebel Login Name: sadmin
Login Name: operations
Password: welcome

7. Go to **Site Map, Administration - Server Configuration - Components**

a. Select the **HTIM Object Manager (ENU)** component. Select the **Order Management - Enable Asset Based Ordering** parameter, and set this value to *False* to invoke Oracle Configurator instead of calling Siebel Configurator.

b. You must activate the following workflows for Oracle Configurator to work. These workflows exist; they were introduced as part of the Oracle Configurator integration:

Account - New Quote
Account - New Order
Goto*
PSP*
Product Compatibility*
Product Eligibility*
Product Reco*
Pric
Compatibility Multiple Popup Workflow Config*
Check*

15.6.2 Copying Oracle Configurator Web Service Setup

To copy Oracle Configurator Data Map setup:

1. Log in to the Siebel HTIM application from a browser.
2. Go to the **Site Map** and select **Administration - Application, Data Map Administration**.

3. For each of the following data maps, add the columns *External Configurator Reference 1, External Configurator Reference 2*, and *External Configurator Reference 3* to both the **Source** column and the **Destination** column in the Data Map Field section of each.

   Data Map Component specified in parentheses:

   - AutoAgreement(Data Map Component:Line Item)
   - AutoAgreeFromOrder(Data Map Component Name:Line Items)
   - CopyOrder(Data Map Component Name:Line Item)
   - CopyQuote(Data Map Component Name:Line Item)
   - OrderToTemplate(Data Map Component Name:Line Item)
   - QuoteToSalesOrder(Data Map Component Name:Line Item)
   - QuoteToServiceOrder(Data Map Component Name:Line Item)
   - QuoteToTemplate(Data Map Component Name:Line Item)
   - ReviseAgreement(Data Map Component Name:Line Item)
   - ReviseOrder(Data Map Component Name:Line Item)
   - ReviseQuote(Data Map Component Name:Line Item)
   - TemplateToOrder(Data Map Component Name:Line Item)
   - TemplateToQuote(Data Map Component Name:Line Item)

15.6.2.1 Setting Up the DoCompression Parameter

To set up the DoCompression parameter:

1. Stop the web server (that is, IIS) in the Siebel environment.
2. Back up the file D:\19924\eappweb\bin\eapps.cfg on the web server.

   Note: Folder D:\19924 is an example. Ensure that you refer to the eapps.cfg file in your installation.

3. Edit eapps.cfg.
4. Reset the DoCompression parameter in the [defaults] section.

   It should now be `DoCompression=FALSE`.

5. Restart the web server.

15.6.2.2 Adding Siebel Custom Applications to Oracle Applications

Create several custom Oracle Applications to allow models to be accessed from Siebel CRM. These custom applications are used during the publication phase of the Oracle Configurator Model development cycle.

To add Siebel custom applications to Oracle applications:

1. Log in to Oracle Applications with the system administrator credentials.
2. Select **Application, Register**.

   Forms start.

3. The Applications - Register form should appear.

   If it does not, navigate to it.

4. Create three new entries, each with four values:

   a. Siebel Quote Integration |SEBLQ | DUMMY_TOP | Provides integration between Siebel Quote and Oracle Configurator

   b. Siebel Order Entry Integration | SEBLO | DUMMY_TOP | Provides integration between Siebel Order Entry and Oracle Configurator

   c. Siebel Agreement Integration | SEBLA | DUMMY_TOP | Provides integration between Siebel Agreement and Oracle Configurator

   Add the new applications to the Oracle Configurator Publication Applicability list.

5. Change responsibility to *Configurator Administrator*.

6. Select **Application to Publication Applicability List** and add the applications.

   This step runs a concurrent program. When the program runs, these new applications are available during model publication.

For more information about the process of developing models for calling applications, see the Oracle Configurator product documentation.

15.6.2.3 Enabling Siebel Eligibility and Compatibility in Oracle Configurator

By default, Siebel Eligibility and Compatibility are not enabled when you launch Oracle Configurator from the Siebel CRM applications listed in the previous section.

To enable Eligibility and Compatibility in Oracle Configurator:

1. Modify the UI Definition settings for the UI of the configuration model to display both list and selling prices.

   For details, see the *Oracle Configurator Developer User's Guide*.

2. Edit the servlet configuration file cz_init.txt and set the servlet property cz.activemodel as follows:

   ```
   cz.activemodel=/lp|/dp|/ec|
   ```

3. If you previously enabled *Eligibility and Compatibility* and want to disable the functionality, set the cz.activemodel servlet property as follows:

   ```
   cz.activemodel=/nolp|/nodp|/noec|
   ```

   The pricing settings in the UI Definition do not have to be changed to disable *Eligibility and Compatibility*.

4. Specify how you want Oracle Configurator to process *Eligibility and Compatibility* violations.

   Do this using either of the following cz.activemodel values:

   - ecv - Display violations as a validation failure
   - noecv - Display violations as an informational message

   For example,

   ```
   cz.activemodel=/lp|/dp|/ec|/ecv|
   ```

   In this example, list and selling (net) prices are enabled, *Eligibility and Compatibility* are enabled, and violations are displayed as validation failures.

For more information about the cz.activemodel servlet property and the cz_init.txt file, see the *Oracle Configurator Installation Guide*.

15.6.2.4 Setting Up Custom Look and Feel

To make the Oracle user interface look like the Siebel user interface:

1. Log in to the Oracle application and select *Application Developer*.
2. Select *Application: Lookups Application Object Library Function*.
3. Query for %LOOK%FEEL%
4. Add a new lookup value with the following entries in a new row for name, meaning, and description:
   - NAME: ABS-DESKTOP
   - MEANING: ABS Custom Look And Feel
   - DESCRIPTION: ABS Custom Look And Feel
5. Save the record and quit.
6. Change responsibility to *System Administrator*.
7. Select the **Oracle Applications Look and Feel** profile and apply the new look and feel by changing the profile value at the user or application level.

15.6.2.5 Displaying Images and Icons from Oracle Configurator in Siebel CRM

To solve problems viewing images, icons, and so forth from Oracle Configurator in Siebel CRM:

1. Create an Apache alias to point to the cabo folder under $OA_HTML.
2. In the Oracle EBS server, modify the Apache configuration file named "apps.conf" by adding the following lines to the file:

```
Alias /cabo/      "/slot/ems1426/appmgr/xz1st102comn/html/cabo/"
<Location /cabo/>
..Order allow,deny
..Allow from all
</Location>
```

3. Restart the Oracle EBS Apache server.
4. Verify that the setup on the Oracle EBS server is correct:
   a. Log on to Oracle EBS.
   b. Add or navigate to the Responsibility Customizing Look and Feel Administrator.
   c. On the Look and Feel Configuration page, select the **Update Look and Feel** option.
   d. Verify that the Look and Feel Name = abs-desktop.
   e. Select **Next**.
   f. On the Customize Styles and Icons page, select the **Grid** component.

      The correct images should appear on this page.
5. Clear the browser cache before performing any test on your computer.

15.7 Working with Cross-References

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

For more information about cross-references, see *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*, "Working with Cross References."

Table 15–3 lists the Order to Cash cross-references:

Table 15–3 Order to Cash Cross-References

Table	Column	Description	Usage
CUSTOMERPARTY_PARTYID	EBIZ_01	Customer party IDs	Lookup during sales order flow, lookup/populated during customer flow
COMMON	Populated by sales order/customer flow	NA	NA
SEBL_01		NA	NA
CUSTOMERPARTY_ ACCOUNTID	EBIZ_01	Customer account IDs	Lookup during sales order flow, lookup/populated during customer flow
COMMON	Populated by sales order/customer flow	NA	NA
SEBL_01	Lookup during asset flow	NA	NA
CUSTOMERPARTY_ ADDRESSID	EBIZ_01	Address (location) IDs	Lookup during sales order flow, lookup/populated during customer flow
COMMON	Populated by sales order/customer flow	NA	NA
SEBL_01	NA	NA	NA
CUSTOMERPARTY_ CONTACTID	EBIZ_01	Contact/Person IDs	Lookup during sales order flow, lookup/populated during customer flow
COMMON	Populated by sales order/customer flow	NA	NA
SEBL_01	NA	NA	NA
CUSTOMERPARTY_ LOCATIONREFID	EBIZ_01	Address (location) IDs	Lookup during sales order flow, lookup/populated during customer flow
COMMON	Populated by sales order/customer flow	NA	NA
SEBL_01	NA	NA	NA
CUSTOMERPARTY_ PARTYLOCATIONID	EBIZ_01	Party address IDs	Lookup during sales order flow, lookup/populated during customer flow
COMMON	Populated by sales order/customer flow	NA	NA
SEBL_01	NA	NA	NA

Table 15–3 (Cont.) Order to Cash Cross-References

Table	Column	Description	Usage
CUSTOMERPARTY_ ACCOUNT_PHONECOMMID	EBIZ_01	Account Phone contact points	lookup/populated during customer flow
COMMON	NA	NA	NA
SEBL_01	NA	NA	NA
CUSTOMERPARTY_ ACCOUNT_FAXCOMMID	EBIZ_01 column	Account Fax contact points	lookup/populated during customer flow
COMMON	NA	NA	NA
SEBL_01	NA	NA	NA
CUSTOMERPARTY_ ACCOUNT_WEBCOMMID	EBIZ_01 column	Account Email/Web contact points	lookup/populated during customer flow
COMMON	NA	NA	NA
SEBL_01	NA	NA	NA
CUSTOMERPARTY_ CONTACT_PHONECOMMID	EBIZ_01 column	Contact's Phone contact points	lookup/populated during customer flow
COMMON	NA	NA	NA
SEBL_01	NA	NA	NA
CUSTOMERPARTY_ CONTACT_FAXCOMMID	EBIZ_01	Contact's Fax contact points	lookup/populated during customer flow
COMMON	NA	NA	NA
SEBL_01	NA	NA	NA
CUSTOMERPARTY_ CONTACT_EMAILCOMMID	EBIZ_01	Contact's Email/Web contact points	lookup/populated during customer flow
COMMON	NA	NA	NA
SEBL_01	NA	NA	NA
CUSTOMERPARTY_ PARTYCONTACTID	EBIZ_01	Party contact IDs	Lookup during sales order flow, lookup/populated during customer flow
COMMON	NA	NA	NA
NA			
SEBL_01	NA	NA	NA
SALESORDER_ID	EBIZ_01	Sales Order ID	Populated by sales order flow
COMMON	NA	NA	NA
SEBL_01	NA	NA	NA
SALESORDER_LINEID	EBIZ_01	Sales Order Line ID	Populated by sales order flow
COMMON	NA	NA	NA
SEBL_01	NA	NA	NA
QUOTE_ID	EBIZ_01	Quote ID	Populated by sales order flow
COMMON	NA	NA	NA
SEBL_01	NA	NA	NA
QUOTE_LINEID	EBIZ_01	Quote Line ID	Populated by sales order flow

Table 15–3 (Cont.) Order to Cash Cross-References

Table	Column	Description	Usage
COMMON	NA	NA	NA
SEBL_01	NA	NA	NA
ORGANIZATION_ID	EBIZ_01	Organization/Business Unit IDs	Lookup during sales order/customer flow
COMMON	Lookup during asset flow	NA	NA
SEBL_01	NA	NA	NA
PRICELIST_ID	EBIZ_01	Pricelist ID	Lookup during sales order flow
COMMON	Populated by Price List bulk load	NA	NA
SEBL_01	NA	NA	NA
ITEM_ITEMID	EBIZ_01	Item/Product ID	Lookup during sales order flow
COMMON	Lookup during asset flow	NA	NA
SEBL_01	NA	NA	NA
INSTALLEDPRODUCT_ID	EBIZ_01, COMMON, SEBL_01	Asset ID	Populated by asset BPEL flow & bulk load
INVENTORY_LOCATION_ID	EBIZ_01,COMMON,SEBL_01	Inventory Location IDs	Lookup by sync product & complex product flow & asset flow
USER_NAME	EBIZ_01,COMMON,SEBL_01	User names	Lookup used by security context services

15.8 Working with DVMs

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

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

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

Table 15–4 lists the DVMs for the Order to Cash: Siebel CRM - EBS integration:

Table 15–4 DVMs for Order to Cash: Siebel CRM - EBS

DVM Type	DVM Column Name	Comments
ADDRESS_COUNTRYID	EBIZ_01, COMMON, SEBL_01	Country codes
STATE	EBIZ_01, COMMON, SEBL_01	State codes

Table 15–4 (Cont.) DVMs for Order to Cash: Siebel CRM - EBS

DVM Type	DVM Column Name	Comments
PROVINCE	EBIZ_01, COMMON, SEBL_01	Province codes
ADDRESS_COUNTRYSUBDIVID	EBIZ_01, COMMON, SEBL_01	Subdivision codes
ORDER_SOURCE_ID	COMMON, EBIZ_01	Mapping for multiple sources of Order/Quote To import order sources and make them AIA-enabled, refer to Ebiz R12 Order Management Guide.
ATPCHECK_STATUS	EBIZ_01, COMMON, SEBL_01	Mapping of E-Business Suite ATP check codes to display messages
CONTACT_SALUTATION	EBIZ_01, COMMON, SEBL_01	Salutation (such as, Mr., Mrs., and so on)
CUSTOMERPARTY_STATUSCODE	EBIZ_01, COMMON, SEBL_01	Account status codes
PHONENUMBER_TYPE	EBIZ_01, COMMON, SEBL_01	Phone number type codes (such as home, work, mobile, fax, and so on)
CONTACT_GENDERCODE	EBIZ_01, COMMON, SEBL_01	Gender code
PHONENUMBER_PURPOSE	EBIZ_01, COMMON, SEBL_01	Phone number purpose (such as personal, business, and so on)
SITEUSAGE_CODE	EBIZ_01, COMMON, SEBL_01	Address site usage code (such as bill to, ship to, and so on)
ITEM_TYPE	EBIZ_01, COMMON, SEBL_01	Mapping of E-Business Suite Item Types to PIM or Siebel Product Types
ITEM_BOM_ITEMTYPE_CODE	EBIZ_01, COMMON, SEBL_01, PIM_01	Mapping of E-Business Suite BOM item type to Siebel structure types
ORACLE_RESPONSIBILITY	EBIZ_01, COMMON, SEBL_01	Mapping of Oracle context/organization to responsibility
UNIT_OF_MEASURE	EBIZ_01, COMMON, SEBL_01	Unit of measure codes
INSTALLEDPRODUCT_STATUS	EBIZ_01, COMMON, SEBL_01	Asset status codes
FREIGHT_TERMS_CODE	EBIZ_01, COMMON, SEBL_01	Freight terms
SHIPPING_PRIORITY	EBIZ_01, COMMON, SEBL_01	Shipping priority codes (such as 2 Day Service)
SHIPPING_METHOD	EBIZ_01, COMMON, SEBL_01	Shipping method codes
CURRENCY_CODE	EBIZ_01, COMMON, SEBL_01	Currency codes

Table 15–4 (Cont.) DVMs for Order to Cash: Siebel CRM - EBS

DVM Type	DVM Column Name	Comments
ORDER_CARRIER_TYPE_CODE	EBIZ_01, COMMON, SEBL_01,GLOG_01	Carrier codes (such as FedEx, UPS, and so on)
ORDER_STATUS	EBIZ_01, COMMON, SEBL_01	Order and quote status codes
ORDER_TYPE	EBIZ_01, COMMON, SEBL_01	Type of order (such as sales order, RMA, quote, and so on)
PAYMENT_TERM	EBIZ_01, COMMON, SEBL_01	Terms of payment for an order (for example, Net30)
ORDER_RETURN_REASON	EBIZ_01, COMMON, SEBL_01	RMA return reason
CREDITCARD_TYPE	EBIZ_01, COMMON, SEBL_01	Credit card type codes (such as Visa, MC, and so on)
ORDER_INTEGRATION_STATUS	COMMON, SEBL_01	In sales order flow, integration status setting of the Siebel order or quote
PRODUCT_STATUS	EBIZ_01, COMMON, SEBL_01	Product status codes
CREDITCHECK_STATUS	EBIZ_01, COMMON, SEBL_01	Credit check code mapping (such as pass, fail)
CREDITCARDAUTH_STATUS	EBIZ_01, COMMON, SEBL_01	Credit card authorization code mapping (such as approved, rejected)
CREDITCARD_EXPIRATIONMONTH	EBIZ_01, COMMON, SEBL_01	Mapping of E-Business Suite months to Siebel months
INSTALLEDPRODUCT_TYPE	COMMON, EBIZ_01	Instance Type Code mapping of E-Business Suite
ORDER_FREIGHT_TERMS_CODE	EBIZ_01, COMMON, SEBL_01	Freight terms codes
ORDER_CHANGE_REASON	EBIZ_01, COMMON, SEBL_01	Holds the possible reasons that are applicable when an order is revised or canceled

Mapping of Shipping Method Code

In Oracle EBS, shipping method codes can be created as a combination of various keys such as Carrier Codes, Modes, Priority (service). Siebel CRM has Carrier Codes and Shipping Method (Shipping Priority Codes), which get stored in the ORDER_CARIIER_TYPE_CODE and SHIPPING_PRIORITY DVMs respectively. DVM SHIPPING_METHOD stores all Oracle EBS codes (such as 000001_Federal Ex_A_2F, which has meaning Federal Express-Air-FedEx 2Day) and corresponding COMMON values. To map from Siebel CRM to Oracle EBS Shipping method codes, SHIPPING_METHOD DVM stores COMMON values that can be in either of the following formats according to the way a shipping method code has been created in Oracle EBS application:

- COMMON value of 'Carrier Type Code' DVM: COMMON value of 'Shipping Priority' DVM (such as FEDEX:2DAYSERVICE
- Only COMMON value of 'Carrier Type Code' DVM (for Oracle EBS Shipping Method Code such as 'UPS')

- Only COMMON value of 'Shipping Priority' DVM (for Oracle EBS Shipping Method Code such as 'Overnight')

In the Oracle AIA Order to Cash, the order flows use the logic according to the mapped Shipping Method Code from Siebel CRM to Oracle EBS based on the values provided for Carrier Code and Shipping priority (Shipping Method) and vice versa from Oracle EBS to Siebel CRM.

Mapping ATP Check Status Codes to Messages

In Oracle EBS, the result of the Available to Promise (ATP) Check is communicated back as a status code. To map these status codes to appropriate error messages, mappings must be established in the ATPCHECK_STATUS DVM. As delivered, mappings for the most common status codes are prepopulated.

The mapping that exists in Oracle EBS can be looked up in the Oracle EBS Lookup type MTL_DEMAND_INTERFACE_ERRORS.

15.9 Enabling Oracle EBS Events

Log in to the Oracle EBS Workflow Administrator responsibility and ensure that the following Business events and the corresponding subscriptions are enabled:

- oracle.apps.csi.instance.update
- oracle.apps.csi.instance.create
- oracle.apps.ont.genesis.outbound.update
- oracle.apps.ego.item.postItemUpdate
- oracle.apps.ego.item.postItemCreate
- oracle.apps.ego.item.postItemBulkload
- oracle.apps.ar.hz.OrgCustBO.update
- oracle.apps.ar.hz.OrgCustBO.create
- oracle.apps.ar.hz.Party.merge
- oracle.apps.ar.hz.CustAccount.merge

15.10 Creating Oracle EBS System Profiles

You create specific profile options for the Customer and Asset Management process integrations.

15.10.1 Creating System Profile Values for the Customer Management Integration

To set specific profile options for the Customer Management integration:

1. Log in to Oracle EBS using the system administrator responsibility.
2. Open the **System Profile Values** form.
3. Query these profile options and set the indicated values at the site level:
   - HZ: Execute API Callouts to All Events Enabled or Only Business Object Events Enabled (applicable to Oracle EBS 11.5.10.2)
   - HZ: Raise API Events to All Events Enabled or Only Business Object Events Enabled (applicable to Oracle EBS 12.1.1)

- HZ: Format Business Object Business Events as Bulk to *N*
- HZ: Generate Party Number to *Yes*
- HZ: Generate Party Site Number to *Yes*

15.10.2 Configuring Receivables System Options for the Customer Management Integration

To configure receivables system options for the Customer Management integration:

1. Log in to Oracle EBS using the Receivables responsibility of the operating unit where customer records are being synced.
2. Open the **System Options** form under **Setup, System** and select the Trans and Customers tab.
3. Select the **Automatic Customer Numbering** and **Automatic Site Numbering** check boxes.

15.10.3 Creating System Profile Values for the Asset Management Integration

To set specific profile options for the Asset Management integration:

1. Log in to Oracle EBS using the System Administrator responsibility.
2. Open the **System Profile Values** form.
3. Query these profile options and set the indicated values at the site level:
   - CSI: Raise Business Event For Customer Owned Instances to *Yes*
   - CSI: Auto-split Instances During Instantiation to *No*

15.10.4 Configuring Oracle EBS for Constraints

To configure Oracle EBS for constraints:

1. Navigate through **Setup, Rules, Security, Processing Constraints**.
2. Query for Order Management Application and Order Line entity in the **Processing Constraints** form.
3. Query for the constraint *Delete of Line Not Allowed* when:
   - Order line is Booked and order line is not an RLM Line
   - Order line is PO Approved
4. Clear the **Constraint** check box.
5. Save and close the form.

15.10.5 Creating Profile Values for the Order Management Integration

To create profile values for the Order Management integration post installation:

1. Log in to the Oracle EBS application and navigate to **System Administrator**. Select **Profile, System**.

2. Query for the profile OM: Create Account Information and change the site value to *All*.
3. Query for the following profiles and set the value to *Yes* at site level:
   - OM: Roll Up Charges on Header Level for AIA Synch
   - OM: Roll Up Charges on Line Level for AIA Synch
   - OM: Roll Up Tax on Header Level for AIA Synch
4. Log in to the Oracle EBS application and navigate to **Order Management Super User Responsibility**.
5. Launch the **Setup: System Parameters - Values** form.
6. Change the value of the Customer Relationships parameter to *All Customers*.

To preserve a submitted selling price for an order line:

1. Define a line level, manual, overrideable, amount-pricing modifier (in Oracle Advanced Pricing setup).
2. Set the value of the system profile option OM: Price Adjustment Modifier for Oracle AIA Order Lines to the modifier defined in the step 1, at the site level.

15.11 Scheduling Concurrent Processes

You must schedule concurrent processes for the Customer Management integration.

For Trading Community Architecture (TCA), a concurrent program must be run to raise business events after data creation. Schedule the concurrent request TCA Business Object Events: Raise Events to run periodically. You can run this request manually.

Before running the TCA Business Object Events: Raise Events concurrent request for the very first time, ensure that the concurrent request TCA Business Object Events: Generate Infrastructure Programs is run first.

Schedule the concurrent request TCA Business Object Events: Cleanse Infrastructure program clean up to run once a day to purge the tracking table.

15.12 Setting a Property in OTM

For the Shipping Charges integration to work successfully, you must set the following property in Oracle Transportation Management 5.5 CU5:

```
glog.integration.remoteQuery.wrapReplyInTransmission=1
```

For more information about the Shipping Charges integration, see Chapter 8, "Shipping Charges Integration Flow."

15.13 Configuring the Payment Authorization Integration

To configure the Payment Authorization integration:

1. Set IBY: ECAPP URL (IBY profile name) to:

   http:// <host machine name>:<port number>/oa_servlets/ibyecapp

   a. Log in to the Oracle EBS application and navigate to **System Administrator, Profile, System**

b. Query for IBY: ECAPP URL and change the value of:

"http://<host machine name>:<port number>/oa_servlets/ibyecapp"

2. Set the servlet base URL to:

http:// <host machine name>:<port number>/oa_servlets

3. Add the responsibility for the user (SYSADMIN).

Navigate to **System Administrator, Security: User, Define**, and query for User (SYSADMIN), and add the responsibility *IPayment Payment Administrator*.

4. Navigate to **IPayment Payment Administrator, iPayment Administrator**.

In iPayment, navigate to the Setup tab, Payment Systems tab, click the update icon of the SamplePaymentSystem and change the value for Servlet base URL to: http:// <host machine name>:<port number>/oa_servlets

For more information about setting up Oracle Order Management transaction types and sequence assignments, see *Oracle EBS Order Management User Guide*, "Simple Negotiation in Oracle Order Management."

15.14 Handling Errors

Based on the roles defined for the services, the system sends email notifications if a service ends in error.

Table 15–5 lists the errors caused by the Order to Cash services.

Table 15–5 Errors Caused by Order to Cash Services

Error Code	Message Text
AIA_ERR_AIAO2C2_1001	Timeout while waiting for a response from the InterfaceCustomerToFulfillmentEBF service.
AIA_ERR_AIAO2C2_1002	Timeout while waiting for a response from the InterfaceSalesOrderToCustomerEBFV2 service.
AIA_ERR_AIAO2C2_1003	Timeout while waiting for a response from the CreateSalesOrder EBS service operation.
AIA_ERR_AIAO2C2_1004	Timeout while waiting for a response from the UpdateSalesOrder EBS service operation.
AIA_ERR_AIAO2C2_1005	This order has been synchronized.
AIA_ERR_AIAO2C2_1006	Account could not be queried. Ensure that the account exists in the system, or review the setup.
AIA_ERR_AIAO2C2_1007	Timeout while waiting for a response from the SyncCustomerPartyList EBS service operation.
AIA_ERR_AIAO2C2_1008	Credit Check failed. Please contact your System Administrator.
AIA_ERR_AIAO2C2_1009	Timeout occurred in UpdateItemInstanceEbizReqABCSImpl waiting for item instance creation

For more information about the errors caused by Siebel CRM or Oracle EBS, see the documentation for that product.

For more information about Oracle AIA error handling, see the *Oracle Fusion Middleware Infrastructure Components and Utilities User's Guide for Oracle Application Integration Architecture Foundation Pack*, "Setting Up Error Handling."

15.14.1 Describing Delivered Error Notification Roles and Users

The system delivers the following roles and users as default values for issuing error notifications for the Order to Cash integration.

Actor Roles and Users

- **Role:** OracleSiebelAdmin User: OracleSiebelAdminUser
- **Role:** AIAIntegrationAdmin User: AIAIntegrationAdminUser

FYI Roles and Users

- **Role:** OracleSiebelCSR User: OracleSiebelCSRUser

The default password set for all users is *welcome1*.

For more information about setting up error notifications using these values, see *Oracle Fusion Middleware Infrastructure Components and Utilities User's Guide for Oracle Application Integration Architecture Foundation Pack*, "Introduction to Oracle AIA Error Handling" and "Using Trace and Error Logs."

15.15 Viewing EBO Implementation Maps (EIMs)

For more information about how services are mapped, see the My Oracle Support document: *EBO Implementation Maps (EIMs)* 1095494.1.

15.16 Setting Configuration Properties

Set these properties in the AIAConfigurationProperties.xml file. The file is located in AIA_HOME>/aia_instances/$INSTANCE_NAME/AIAMetaData/config/.

For more information about requirements for working with AIAConfigurationProperties.xml, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Building AIA Integration Flows," How to Set Up AIA Workstation.

Table 15–6 shows settings for the system properties:

Table 15–6 Settings: System Properties

System Property	Value/Default Value	Description
O2C.EnablePriceListMapping	true/false Default = false	Enable/disable mapping and validation of pricelist during order & quote processing.

Table 15–7 shows settings for the module level properties.

Table 15–7 Settings: Module -Level Properties

Module	Property	Value/Default Value	Description
Siebel	SEBL_01.SERVER_TIMEZONE	GMT-08:00	Time zone code of Siebel Server
Siebel	SEBL_01.UTCCanonicalFlag	false	Determines when the date format is in UTC or not. Set this to true if UTC is enabled on the Siebel Application. Else set to false.
Siebel	ComplexXrefKeyDelimiter	:	Delimiter for concatenating Ids in XREF data. Set this to any character (non-system type) that is not used for creating Ids in the Siebel Application.
BULKLOAD	BULKLOAD.DEFAULT.SOURCE	No default value	Ebiz system code (defined in OER), which is the source application for initial loads
BULKLOAD.DEFAULT.TARGET	No default value	Siebel system code (defined in OER), which is the target application for initial loads	NA
Ebiz	EBIZ_01.SERVER_TIMEZONE	No default value	Time zone code

Table 15–8shows settings for the InterfaceCustomerToFulfillmentEBF service property:

Table 15–8 Settings: InterfaceCustomerToFulfillmentEBF Service Property

Property Name	Value/Default Value	Description
SyncCustomerPartyListResponseRequired	true/false, Default = true	Standard properties to control EBF functionality
InterfaceCustomerToFulfillment	true/false Default = true	Standard properties to control EBF functionality
ProcessCustomerPartyList	true/false Default = true	Standard properties to control EBF functionality
SyncCustomerPartyList.AsyncTimeoutDuration	Default value = PT5M	Specifies the time for which the service waits to receive a response. If response is not received within this time, the process times out and terminates.
Routing.CustomerPartyEBSV2.SyncCustomerPartyList.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
Routing.CustomerPartyEBSV2.QueryCustomerPartyList.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
Routing.CustomerPartyOrchestrationResponseEBSV2.InterfaceCustomerToFulfillmentResponse.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.

Table 15–8 (Cont.) Settings: InterfaceCustomerToFulfillmentEBF Service Property

Property Name	Value/Default Value	Description
Routing.CustomerPartyOrchestrationResponseEBSV2.InterfaceCustomerToFulfillmentResponse.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.CustomerPartyEBSV2.QueryCustomerPartyList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SalesOrderOrchestrationResponseEBSV2.InterfaceCustomerToFulfillmentResponse.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.CustomerPartyEBSV2.SyncCustomerPartyList.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.CustomerPartyEBSV2.QueryCustomerPartyList.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.CustomerPartyOrchestrationResponseEBSV2.InterfaceCustomerToFulfillmentResponse.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncrequestrecipient	CAVS Endpoint URI, when CAVS is enabled.

Table 15–9shows settings for the SyncAccountSiebelReqABCSImpl service property:

Table 15–9 Settings: SyncAccountSiebelReqABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	No default value	Siebel system code (like SEBL_01, defined in OER) from which requests originate for this process
Routing.CustomerPartyEBSV2.SyncCustomerPartyList.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
Routing.CustomerPartyEBSV2.SyncCustomerPartyList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.CustomerPartyEBSV2.SyncCustomerPartyList.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncrequestrecipient	CAVS Endpoint URI, when CAVS is enabled.

Table 15–9 (Cont.) Settings: SyncAccountSiebelReqABCSImpl Service Property

Property Name	Value/Default Value	Description
ABCSExtension.PreXformABMtoEBMABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeEBSEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
Account.ProcessUpdateEventsOnly	True	This property governs whether create events raised in the Siebel application are consumed. By default, only update events are consumed.
Routing.SWICustomerParty.SEBL_01.EndpointURI	http://<SIEBEL_HOST>/eai_enu/start.swe?SWEExtSource=WebService&SWEExtCmd=Execute&WSSOAP=1	Siebel SWICustomerPartyService (Query) service endpoint location. This is a SOAP endpoint URL. If the request message contains the target URL, then that takes precedence.
Routing.SWICustomerParty.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SWICustomerParty.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.

Table 15–10 shows settings for the SyncCustomerPartyListEbizReqABCSImpl service property:

Table 15–10 Settings: SyncCustomerPartyListEbizReqABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	No default value	Ebiz system code (like EBIZ_01, defined in OER) from which requests originate for this process
Routing.QueryCustomerPartyListEbizCreate.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.QueryCustomerPartyListEbizCreate.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property
Routing.QueryCustomerPartyListEbizCreate.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/QueryCustomerPartyListEbizCreateAdapter/QueryCustomerPartyListEbizCreateAdapter_ep	Endpoint URI of the Ebiz adapter

Table 15–10 (Cont.) Settings: SyncCustomerPartyListEbizReqABCSImpl Service Property

Property Name	Value/Default Value	Description
Routing.QueryCustomerPartyListEbizUpdate.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.QueryCustomerPartyListEbizUpdate.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.QueryCustomerPartyListEbizUpdate.EBIZ_01.EndpointURI r	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/QueryCustomerPartyListEbizUpdateAdapter/QueryCustomerPartyListEbizUpdateAdapter_ep	Endpoint URI of the Ebiz adapter
Routing.CustomerPartyEBSV2.SyncCustomerPartyList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.CustomerPartyEBSV2.SyncCustomerPartyList.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
Routing.CustomerPartyEBSV2.SyncCustomerPartyList.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncrequestrecipient	CAVS Endpoint URI, when CAVS is enabled.
ABCSExtension.PreXformCreateABMtoEBMABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreXformUpdateABMtoEBMABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeCustEBSEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–11 shows settings for the QueryCustomerPartyListSiebelProvABCSImplV2 service property:

Table 15–11 Settings: QueryCustomerPartyListSiebelProvABCSImplV2 Service Property

Property Name	Value/Default Value	Description
Default.SystemID	No default value	Siebel system code (such as SEBL_01, defined in OER) from which data is queried by this process
Routing.SWI_spcCustomer_spcParty_spcService.SEBL_01.EndpointURI	http://<SIEBEL_HOST>/eai_enu/start.swe?SWEExtSource=WebService&SWEExtCmd=Execute&WSSOAP=1	Siebel SWICustomerPartyService (Query) service endpoint location. This is a SOAP endpoint URL. If the request message contains the target URL, then that takes precedence.
Routing.SWI_spcCustomer_spcParty_spcService.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SWI_spcCustomer_spcParty_spcService.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
ABCSExtension.PreXformEBMtoABMEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeSWICustServiceABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostInvokeSWICustServiceABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostXformABMtoEBMEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–12 shows settings for the SyncCustomerPartyListEbizProvABCSImpl service property:

Table 15–12 Settings: SyncCustomerPartyListEbizProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	No default value	Ebiz system code (such as EBIZ_01, defined in OER) to which data is synced. The System code value present in the enterprise business message (EBM) header of the incoming message takes precedence over this property value.
Routing.SyncCustomerPartyListEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/SyncCustomerPartyListEbizAdapter/SyncCustomerPartyListEbizAdapter_ep	Endpoint URI of the Ebiz adapter
Routing.SyncCustomerPartyListEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.SyncCustomerPartyListEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.CustomerPartyResponseEBSV2.SyncCustomerPartyListResponse.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
Routing.CustomerPartyResponseEBSV2.SyncCustomerPartyListResponse.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.CustomerPartyResponseEBSV2.SyncCustomerPartyListResponse.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncrequestrecipient	CAVS Endpoint URI, when CAVS is enabled.
ABCSExtension.PreXformEBMtoABMEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–12 (Cont.) Settings: SyncCustomerPartyListEbizProvABCSImpl Service Property

Property Name	Value/Default Value	Description
ABCSExtension.PreInvokeEbizAdapterABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostInvokeEbizAdapterABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostXformABMtoEBMEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–13 shows settings for the SyncCustomerPartyListSiebelProvABCSImpl service property:

Table 15–13 Settings: SyncCustomerPartyListSiebelProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	No default value	Siebel system code (such as SEBL_01, defined in OER) to which data is synced. The System code value present in the EBM header of the incoming message takes precedence over this property value.
Routing.SyncCustomerPartyListSiebelService.SEBL_01.EndpointURI	http://<SIEBEL_HOST>/eai_enu/start.swe?SWEExtSource=WebService&SWEExtCmd=Execute&WSSOAP=1	Siebel SWICustomerPartyService(Upsert) service endpoint location. This is a SOAP endpoint URL. If the request message contains the target URL, then that takes precedence.
Routing.SyncCustomerPartyListSiebelService.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SyncCustomerPartyListSiebelService.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.MergeCustomerPartyListSiebelService.SEBL_01.EndpointURI	http://<SIEBEL_HOST>/eai_enu/start.swe?SWEExtSource=WebService&SWEExtCmd=Execute&WSSOAP=1	Siebel SWICustomerPartyService(Merge) service endpoint location. This is a SOAP endpoint URL. If the request message contains the target URL, then that takes precedence.

Table 15–13 (Cont.) Settings: SyncCustomerPartyListSiebelProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Routing.MergeCustomerPartyListSiebelService.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.MergeCustomerPartyListSiebelService.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.CustomerPartyEBSV2.SyncCustomerPartyListResponse.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
Routing.CustomerPartyEBSV2.SyncCustomerPartyListResponse.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.CustomerPartyEBSV2.SyncCustomerPartyListResponse.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncrequestrecipient	CAVS Endpoint URI, when CAVS is enabled.
Routing.CustomerPartyResponseEBSV2.SyncCustomerPartyListResponse.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
ABCSExtension.PreXformEBMtoABMEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeSWICustSyncServiceABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeSWICustUpsertServiceABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostInvokeSWICustSyncServiceABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostInvokeSWICustUpsertServiceABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–14 shows settings for the MergeAccountEbizReqABCSImpl service property:

Table 15–14 Settings: MergeAccountEbizReqABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	No default value	Ebiz system code (such as EBIZ_01, defined in OER) from which requests originate for this process.
Routing.QueryMergeAccountEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/QueryMergeAccountEbizAdapter/QueryMergeAccountEbizAdapter_ep	Endpoint URI of the Ebiz adapter.
Routing.QueryMergeAccountEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.QueryMergeAccountEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.QueryCustomerPartyEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/QueryCustomerPartyEbizAdapter/QueryCustomerPartyEbizAdapter_ep	Endpoint URI of the Ebiz adapter.
Routing.QueryCustomerPartyEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.QueryCustomerPartyEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.CustomerPartyEBS.SyncCustomerPartyList.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncrequestrecipient	CAVS Endpoint URI, when CAVS is enabled.
Routing.CustomerPartyEBS.SyncCustomerPartyList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.CustomerPartyEBS.SyncCustomerPartyList.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes to the target service.

Table 15–14 (Cont.) Settings: MergeAccountEbizReqABCSImpl Service Property

Property Name	Value/Default Value	Description
ABCSExtension.PreXformSyncAcctABMToEBMABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreXformMergeAcctABMToEBMABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeCustPartyEBSEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–15 shows settings for the MergePartyEbizReqABCSImpl service property:

Table 15–15 Settings: MergePartyEbizReqABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	No default value	Ebiz system code (such as EBIZ_01, defined in OER) from which requests originate for this process.
Routing.QueryPartyMergeEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/QueryPartyMergeEbizAdapter/QueryPartyMergeEbizAdapter_ep	Endpoint URI of the Ebiz adapter.
Routing.QueryPartyMergeEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.QueryPartyMergeEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.QueryMergeOrgCustEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/QueryMergeOrgCustEbizAdapter/QueryMergeOrgCustEbizAdapter_ep	Endpoint URI of the Ebiz adapter.
Routing.QueryMergeOrgCustEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.QueryMergeOrgCustEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.

Table 15–15 (Cont.) Settings: MergePartyEbizReqABCSImpl Service Property

Property Name	Value/Default Value	Description
Routing.QueryRelatedOrgCustEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/QueryRelatedOrgCustEbizAdapter/QueryRelatedOrgCustEbizAdapter_ep	Endpoint URI of the Ebiz adapter.
Routing.QueryRelatedOrgCustEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.QueryRelatedOrgCustEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.CustomerPartyEBS.SyncCustomerPartyList.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncrequestrecipient	CAVS Endpoint URI, when CAVS is enabled.
Routing.CustomerPartyEBS.SyncCustomerPartyList.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
Routing.CustomerPartyEBS.SyncCustomerPartyList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
ABCSExtension.PreXformABMToEBMABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreXformRelatedABMToEBMABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeSyncCustomerPartyEBSEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–16 shows settings for the InterfaceSalesOrderToFulfillmentEBF service property:

Table 15–16 Settings: InterfaceSalesOrderToFulfillmentEBF Service Property

Property Name	Value/Default Value	Description
InterfaceSalesOrderToCustomerResponseRequired	true/false Default = true	Standard properties to control EBF functionality.
Routing.SalesOrderEBSV2.SyncSalesOrderList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS EndpointURI property.
Routing.SalesOrderEBSV2.SyncSalesOrderList.CAVS.EndpointURI	true/false Default = false	CAVS Endpoint URI, when CAVS is enabled.
isLegacyEbizProviderSupported	true/false Default = false	Governs whether the SyncSalesOrderEbizProvABCSImpl or CreateSalesOrderEbizProvABCSImpl is used. Default value is false, which uses SyncSalesOrderEbizProvABCSImpl service.
InterfaceSalesOrderToCustomer	True	Controls whether the InterfaceSalesOrderToCustomerEBF is invoked during order processing.
UpdateSalesOrder	True	Governs whether Update Order operation has to perform to update the sales order back in the source system during order processing.
RMAOrderTypes	RMA	List of values for RMA order types. During processing of orders with this type, Customer sync does not occur. This is a comma-separated (no spaces) list of string values.
InterfaceSalesOrderToCustomer.AsyncTimeoutDuration	PT5M30S	Specifies the time for which the service waits to receive a response. If response is not received within this time, the process times out and terminates. Duration to wait for the asynchronous callback from the InterfaceSalesOrderToCustomer service. If response is not received within the wait duration, a fault occurs.
CreateSalesOrder.AsyncTimeoutDuration	PT3M	Specifies the time for which the service waits to receive a response. If response is not received within this time, the process times out and terminates. Duration to wait for the asynchronous CreateSalesOrderResponse callback from the SalesOrderResponseEBS service. If response is not received within the wait duration, a fault occurs.
UpdateSalesOrder.AsyncTimeoutDuration	PT1M	Specifies the time for which the service waits to receive a response. If response is not received within this time, the process times out and terminates. Duration to wait for the asynchronous UpdateSalesOrderResponse callback from the SalesOrderResponseEBS service. If response is not received within the wait duration, a fault occurs.

Table 15–16 (Cont.) Settings: InterfaceSalesOrderToFulfillmentEBF Service Property

Property Name	Value/Default Value	Description
Routing.SalesOrderEBSV2.CreateSalesOrder.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SalesOrderEBSV2.CreateSalesOrder.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.SalesOrderEBSV2.UpdateSalesOrder.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SalesOrderEBSV2.UpdateSalesOrder.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.SalesOrderOrchestrationEBSV2.InterfaceSalesOrderToCustomer.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI.property.
Routing.SalesOrderOrchestrationEBSV2.InterfaceSalesOrderToCustomer.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.SalesOrderOrchestrationResponseEBSV2.InterfaceSalesOrderToFulfillmentResponse.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SalesOrderOrchestrationResponseEBSV2.InterfaceSalesOrderToFulfillmentResponse.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponserecipient	CAVS Endpoint URI, when CAVS is enabled.

Table 15–17 shows settings for the InterfaceSalesOrderToCustomerEBFV2 service property:

Table 15–17 Settings: InterfaceSalesOrderToCustomerEBFV2 Service Property

Property Name	Value/Default Value	Description
InterfaceCustomerToFulfillment ResponseRequired	true/false Default = false	Standard properties to control EBF functionality.
Routing.CustomerPartyOrchest rationEBSV2.InterfaceCustomer ToFulfillment.RouteTo CAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.CustomerPartyOrchest rationEBSV2.InterfaceCustomer ToFulfillment.CAVS.EndpointU RI	http://<SOA_HOST>:<SOA_ PORT>/AIAValidationSystemServ let/asyncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.SalesOrderOrchestratio nResponseEBSV2.InterfaceSales OrderToCustomerResponse.Ro uteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SalesOrderOrchestratio nResponseEBSV2.InterfaceSales OrderToCustomerResponse.CA VS.EndpointURI	http://<SOA_HOST>:<SOA_ PORT>/AIAValidationSystemServ let/asyncresponserecipient	CAVS Endpoint URI, when CAVS is enabled.
InterfaceCustomerToFulfillment .AsyncTimeoutDuration	PT5M	Specifies the time for which the service waits to receive a response. If response is not received within this time, the process times out and terminates. Duration to wait for the asynchronous InterfaceCustomerToFulfillmentResponse callback from the SalesOrderOrchestrationResponseEBS service. If response is not received within the wait duration, a fault occurs.

Table 15–18 shows settings for the ProcessSalesOrderSiebelReqABCSImplV2 service property:

Table 15–18 Settings: ProcessOrderSiebelReqABCSImplV2 Service Property

Property Name	Value/Default Value	Description
- SEBL_01.ApplicationVersion	$siebel.version (example, 8.1.1.x or 8.0.0.x)	Takes Siebel version values provided while you are installing the integration based on which it calls the respective web service version in Siebel.
Routing.SBLOrderUpsertService.SEBL_01.EndpointURI	Siebel SBLOrderUpsert service endpoint location (example, Siebel version 8.0.7 endpoint location).	This is a SOAP endpoint URL. If the request message contains the target URL, then that takes precedence.
Routing.SBLOrderUpsertService.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SBLOrderUpsertService.CAVS.EndpointURI	true/false Default = false	CAVS Endpoint URI, when CAVS is enabled.
isResponseRequired	true/false Default = true	Governs whether responseCode should be populated in EBM indicating that EBF (example, InterfaceSalesOrderToFulfillment) should return the response.
Default.SystemID	SEBL_01	Siebel system code (such as SEBL_01, defined in OER) from which requests originate for this process.
Routing.SalesOrderOrchestrationEBSV2.ProcessSalesOrder.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SalesOrderOrchestrationEBSV2.ProcessSalesOrder.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncrequestrecipient	CAVS Endpoint URI, when CAVS is enabled.
Routing.SWIOrderUpsert.SEBL_01.EndpointURI	http://<SIEBEL_HOST>/eai_enu/start.swe?SWEExtSource=SecureWebService&SWEExtCmd= Execute&WSSOAP=1	Siebel SWIOrderUpsert service endpoint location. This is a SOAP endpoint URL. If the request message contains the target URL, then that takes precedence.
Routing.SWIOrderUpsert.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SWIOrderUpsert.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.

Table 15–18 (Cont.) Settings: ProcessOrderSiebelReqABCSImplV2 Service Property

Property Name	Value/Default Value	Description
ABCSExtension.PreXformABMtoEBMABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeEBSEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
RMAOrderTypes	RMA	Defines the value for RMA order types. This is a list of code values that represent RMA order type, which affects the attribute mappings. This is a comma-separated (no spaces) list of string values.

Table 15–19 shows settings for the ProcessQuoteSiebelReqABCSImpl service property:

Table 15–19 Settings: ProcessQuoteSiebe;ReqABCSImpl Service Property

Property Name	Value/Default Value	Description
isResponseRequired	true/false Default = false	Governs whether responseCode should be populated in EBM indicating that EBF (example, InterfaceSalesOrderToFulfillment) should return the response.
Default.SystemID	SEBL_01	Siebel system code (such as SEBL_01, defined in OER) from which requests originate for this process.
Routing.SalesOrderOrchestrationEBSV2.ProcessSalesOrder.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SalesOrderOrchestrationEBSV2.ProcessSalesOrder.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncrequestrecipient	CAVS Endpoint URI, when CAVS is enabled.
Routing.SWIQuoteUpsert.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SWIQuoteUpsert.SEBL_01.EndpointURI	http://<SIEBEL_HOST>/eai_enu/start.swe?SWEExtSource=SecureWebService&SWEExtCmd= Execute&WSSOAP=1	Siebel SWIQuoteUpsert service endpoint location. This is a SOAP endpoint URL. If the request message contains the target URL, then that takes precedence.

Table 15–19 (Cont.) Settings: ProcessQuoteSiebe;ReqABCSImpl Service Property

Property Name	Value/Default Value	Description
Routing.SWIQuoteUpsert.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
ABCSExtension.PreXformABMtoEBMABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeEBSEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–20 shows settings for the UpdateSalesOrderEbizReqABCSImpl service property:

Table 15–20 Settings: UpdateSalesOrderEbizReqABCSImpl Service Property

Property Name	Value/Default Value	Description
IsOrderLineShippingDetailsRequired	true/false Default = true	Flag for specifying whether synching shipping details for a order line gets shipped
Routing.ShipmentAdviceEBS.SyncShipmentAdviceList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.ShipmentAdviceEBS.SyncShipmentAdviceList.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncrequestrecipient	CAVS Endpoint URI, when CAVS is enabled.
Routing.GetSalesOrderLineShippingDetailsEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.GetSalesOrderLineShippingDetailsEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/GetSalesOrderLineShippingDetailsEbizAdapter/GetSalesOrderLineShippingDetailsEbizAdapter_ep	Endpoint URI of the Ebiz adapter.
ABCSExtension.PreXformOutputParametersToSyncShipmentAdviceListEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked

Table 15–20 (Cont.) Settings: UpdateSalesOrderEbizReqABCSImpl Service Property

Property Name	Value/Default Value	Description
ABCSExtension.PreInvokeShipmentAdviceEBS	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
Default.SystemID	EBIZ_01	Ebiz system code (such as EBIZ_01, defined in OER) from which requests originate for this process.
Routing.SalesOrderEBSV2.UpdateSalesOrder.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SalesOrderEBSV2.UpdateSalesOrder.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncrequestrecipient	CAVS Endpoint URI, when CAVS is enabled.
Routing.GetSalesOrderEbizAdpater.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.GetSalesOrderEbizAdpater.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/GetSalesOrderEbizAdapter/GetSalesOrderEbizAdapter_ep	Endpoint URI of the Ebiz adapter.
Routing.GetSalesOrderEbizAdpater.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.GetItemValidationOrgEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.GetItemValidationOrgEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/GetItemValidationOrganizationOUEbizAdapter/GetItemValidationOrganizationOUEbizAdapter_ep	Endpoint URI of the Ebiz adapter.
Routing.GetItemValidationOrgEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
XRefCheck.IntervalSleepTimeMilliseconds	5000	Specifies the interval time for which the service sleeps to check whether xref has been populated during create order process.

Table 15–20 (Cont.) Settings: UpdateSalesOrderEbizReqABCSImpl Service Property

Property Name	Value/Default Value	Description
XRefCheck.TotalWaitTime	PT30S	Specifies the total wait time for which the service waits regardless of whether xref has been populated during Create Order process.
ABCSExtension.PreXformOutputParametersToUpdateSalesOrderEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeSalesOrderEBS	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–21shows settings for the UpdateSalesOrderSiebelProvABCSImpl service property:

Table 15–21 Settings: Update SalesOrderSiebelProvABCSImpl Service Property

Property Name	Value/Default Value	Description
SyncNewChildLine	true/false Default = false	To activate or inactivate the behavior that specifies whether to sync new child component lines for example, in case of BOM split child lines or included item line sync.
SEBL_01.ApplicationVersion	$siebel.version (example, 8.1.1.x or 8.0.0.x)	Takes Siebel version values provided while you are installing the integration based on which it calls the respective web service version in Siebel.
Routing.SBLOrderUpsertService.SEBL_01.EndpointURI	http://<SIEBEL_HOST>/eai_enu/start.swe?SWEExtSource=SecureWebService&SWEExtCmd=Execute&WSSOAP=1	Siebel SBLOrderUpsert service endpoint location. This is a SOAP endpoint URL. If the request message contains the target URL, then that takes precedence.
Routing.SBLOrderUpsertService.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SBLOrderUpsertService.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
ABCSExtension.PreInvokeSWIOrderUpsertABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostInvokeSWIOrderUpsertABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–21 (Cont.) Settings: Update SalesOrderSiebelProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	SEBL_01	Siebel system code (such as SEBL_01, defined in OER) from which requests originate for this process.
Routing.SWIOrderUpsertService.SEBL_01.EndpointURI	http://<SIEBEL_HOST>/eai_enu/start.swe?SWEExtSource=SecureWebService&SWEExtCmd=Execute&WSSOAP=1	Siebel SWIOrderUpsert service endpoint location. This is a SOAP endpoint URL. If the request message contains the target URL, then that takes precedence.
Routing.SWIOrderUpsertService.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SWIOrderUpsertService.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.SWIQuoteUpsertService.SEBL_01.EndpointURI	http://<SIEBEL_HOST>/eai_enu/start.swe?SWEExtSource=SecureWebService&SWEExtCmd=Execute&WSSOAP=1	Siebel SWIQuoteUpsert service endpoint location. This is a SOAP endpoint URL. If the request message contains the target URL, then that takes precedence.
Routing.SWIQuoteUpsertService.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SWIQuoteUpsertService.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.SalesOrderResponseEBSV2.UpdateSalesOrderResponse.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SalesOrderResponseEBSV2.UpdateSalesOrderResponse.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponserecipient	CAVS Endpoint URI, when CAVS is enabled.
ABCSExtension.PreXformEBMtoABMEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeSBLOrderUpsertABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeSBLQuoteUpsertABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–21 (Cont.) Settings: Update SalesOrderSiebelProvABCSImpl Service Property

Property Name	Value/Default Value	Description
ABCSExtension.PostInvokeSBL OrderUpsertABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostInvokeSBL QuoteUpsertABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostXformABM toEBMEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–22 shows settings for the CreateSalesOrderEbizProvABCSImpl service property:

Table 15–22 Settings: CreateSalesOrderEbizProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	EBIZ_01	Ebiz system code (such as EBIZ_01, defined in OER) from which requests originate for this process.
Routing.ProcessSalesOrderEbiz Adapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_ PORT>/ soa-infra/services/default/Proc essSalesOrderEbizAdapter/Proc essSalesOrderEbizAdapter_ep	Endpoint URI of the Ebiz adapter.
Routing.ProcessSalesOrderEbiz Adapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.ProcessSalesOrderEbiz Adapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_ PORT>/AIAValidationSystemSe rvlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.SalesOrderResponseE BSV2.CreateSalesOrderRespon se.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SalesOrderResponseE BSV2.CreateSalesOrderRespon se.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_ PORT>/AIAValidationSystemSe rvlet/asyncresponserecipient	CAVS Endpoint URI, when CAVS is enabled.
RMAOrderTypes	RMA	Defines the value for RMA order types. This is a list of code values that represent RMA order type, which affects the attribute mappings. This is a comma-separated (no spaces) list of string values.

Table 15–22 (Cont.) Settings: CreateSalesOrderEbizProvABCSImpl Service Property

Property Name	Value/Default Value	Description
ABCSExtension.PreXformEBMToABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeProcessSOEbizAdapter	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostInvokeProcessSOEbizAdapter	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostXformABMToEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
OrderSourceReference	COMMON/SourceSystemIdentifier Default=COMMON	Property to decide whether OSR should hold the Common ID or Source System Identifier.
DefaultOrderSourceId	28	Property to supply the ORDER_SOURCE_ID if dvm lookup failure occurs for quotes.
DefaultQuoteSourceId	29	Property to support the ORDER_SOURCE_ID if dvm lookup failure occurs for quotes.

Table 15–23 shows settings for the CheckATPSalesOrderSiebelReqABCSImpl service property:

Table 15–23 Settings: CheckATPSalesOrderSiebelReqABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	SEBL_01	Siebel system code (such as SEBL_01, defined in OER) from which requests originate for this process.
Routing.SalesOrderEBS.ProcessSalesOrderATPCheck.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes to the target service.
ABCSExtension.PreXformABMtoEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeEBSCheckATPEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–23 (Cont.) Settings: CheckATPSalesOrderSiebelReqABCSImpl Service Property

Property Name	Value/Default Value	Description
ABCSExtension.PostInvokeEBS CheckATPEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostXformABM toEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
Routing.SalesOrderEBS.ProcessS alesOrderATPCheck.RouteToCA VS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SalesOrderEBS.ProcessS alesOrderATPCheck.CAVS.End pointURI	http://<SOA_HOST>:<SOA_ PORT>/AIAValidationSystemS ervlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.

Table 15–24 shows settings for the PaymentAuthorizationSalesOrderSiebelReqABCSImpl service property:

Table 15–24 Settings: PaymentAuthorizationSalesOrderSiebelReqABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	SEBL_01	Siebel system code (like SEBL_01, defined in OER) from which requests originate for this process.
Routing.ReceivedPaymentEBS.P rocessCreditChargeAuthorizatio n.MessageProcessingInstruction. EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
ABCSExtension.PreXformABMt oEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeEBSP aymentAuthorizationEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostInvokeEBS PaymentAuthorizationEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–24 (Cont.) Settings: PaymentAuthorizationSalesOrderSiebelReqABCSImpl Service Property

Property Name	Value/Default Value	Description
ABCSExtension.PostXformABMtoEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
Routing.ReceivedPaymentEBS.ProcessCreditChargeAuthorization.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.ReceivedPaymentEBS.ProcessCreditChargeAuthorization.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.

Table 15–25 shows settings for the CreditCheckSalesOrderSiebelReqABCSImpl service property:

Table 15–25 Settings: CreditCheckSalesOrderSiebelReqABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	SEBL_01	Siebel system code (such as SEBL_01, defined in OER) from which requests originate for this process.
Routing.CustomerPartyEBSV2.ProcessCreditEligibilityVerification.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
Routing.CustomerPartyEBSV2.ProcessCreditEligibilityVerification.RouteToCAVS	true/false Default = false	CAVS Endpoint URI, when CAVS is enabled.
Routing.CustomerPartyEBSV2.ProcessCreditEligibilityVerification.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
ABCSExtension.PreInvokeEBSCreditCheck	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–25 (Cont.) Settings: CreditCheckSalesOrderSiebelReqABCSImpl Service Property

Property Name	Value/Default Value	Description
ABCSExtension.PreXformABMtoEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostInvokeEBSCreditCheck	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostXformEBMtoABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–26shows settings for the CalculateShippingChargeSalesOrderSiebelReqABCSImpl service property:

Table 15–26 Settings: CalculateShippingChargeSalesOrderSiebelReqABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	SEBL_01	Siebel system code (such as SEBL_01, defined in OER) from which requests originate for this process.
Routing.SalesOrderEBSV2.ProcessSalesOrderShippingChargeCalculation.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
Routing.SalesOrderEBSV2.ProcessSalesOrderShippingChargeCalculation.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SalesOrderEBSV2.ProcessSalesOrderShippingChargeCalculation.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
ABCSExtension.PreInvokeEBSCalculateShippingCharge	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–26 (Cont.) Settings: CalculateShippingChargeSalesOrderSiebelReqABCSImpl Service Property

Property Name	Value/Default Value	Description
ABCSExtension.PreXformABMtoEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostInvokeEBSCalculateShippingCharge	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostXformEBMtoABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–27shows settings for the ProcessSalesOrderATPCheckEbizProvABCSImpl service property:

Table 15–27 Settings: ProcessSalesOrderATPCheckEbizProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	EBIZ_01	Ebiz system code (such as EBIZ_01, defined in OER) from which requests originate for this process.
Routing.ProcessSalesOrderATPCheckEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/ProcessSalesOrderATPCheckEbizAdapter/ProcessSalesOrderATPCheckEbizAdapter_ep	Endpoint URI of the Ebiz adapter.
Routing.ProcessSalesOrderATPCheckEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.ProcessSalesOrderATPCheckEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.FetchATPScheduleSequenceEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/FetchATPScheduleSequenceEbizAdapter/FetchATPScheduleSequenceEbizAdapter_ep	Endpoint URI of the Ebiz adapter.
Routing.FetchATPScheduleSequenceEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.

Table 15–27 (Cont.) Settings: ProcessSalesOrderATPCheckEbizProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Routing.FetchATPScheduleSequenceEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
ABCSExtension.PreXformEBMtoABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeCheckATPEbizAdapterABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostXformABMtoEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostInvokeCheckATPEbizAdapterABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–28 shows settings for the ProcessCreditChargeAuthorizationEbizProvABCSImpl service property:

Table 15–28 Settings: ProcessCreditChargeAuthorizationEbizProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	EBIZ_01	Ebiz system code (such as EBIZ_01, defined in OER) from which requests originate for this process.
Routing.ProcessCreditChargeAuthorizationEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/ProcessCreditChargeAuthorizationEbizAdapter/ProcessCreditChargeAuthorizationEbizAdapter_ep	Endpoint URI of the Ebiz adapter.
Routing.ProcessCreditChargeAuthorizationEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.ProcessCreditChargeAuthorizationEbizAdapter.RouteToCAVS	true/false Default = false	Govern whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.

Table 15–28 (Cont.) Settings: ProcessCreditChargeAuthorizationEbizProvABCSImpl Service Property

Property Name	Value/Default Value	Description
ABCSExtension.PreXformEBMtoABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokePaymentAuthEbizAdapterABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostInvokePaymentAuthEbizAdapterABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostXformABMtoEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–29 shows settings for the ProcessCreditEligibilityEbizProvABCSImpl service property:

Table 15–29 Settings: ProcessCreditEligibilityEbizProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	EBIZ_01	Ebiz system code (such as EBIZ_01, defined in OER) from which requests originate for this process.
Routing.ProcessCreditEligibilityEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/ProcessCreditEligibilityEbizAdapterZ/ProcessCreditEligibilityEbizAdapter_ep	Endpoint URI of the Ebiz adapter.
Routing.ProcessCreditEligibilityEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.ProcessCreditEligibilityEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
ABCSExtension.PreXformEBMtoABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–29 (Cont.) Settings: ProcessCreditEligibilityEbizProvABCSImpl Service Property

Property Name	Value/Default Value	Description
ABCSExtension.PreInvokeProcessCreditEligibilityEbizAdapterABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostXformABMtoEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostInvokeProcessCreditEligibilityEbizAdapterABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–30 shows settings for the ProcessSalesOrderShippingChargeLogisticsProvABCSImpl service property:

Table 15–30 Settings: ProcessSalesOrderShippingChargeLogisticsProvABCSImpl

Property Name	Value/Default Value	Description
Default.SystemID	LOGIS_01	Logistics system code (such as LOGIS_01, defined in OER) from which requests originate for this process.
Routing.GLogService.LOGIS_01.EndpointURI	http://otm-dobson-55-oas.us.xxxxxx.com/GC3Services/IntXmlService/webservice	Endpoint URI of the GLOG service.
Routing.GLogService.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.GLogService.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
ABCSExtension.PreXformEBMtoABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostXformABMtoEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeLogisticsABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostInvokeLogisticsABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
Logistics.CalculateShippingCharge.TransmissionType	QUERY	Specifying Logistics web service. It is a query type. (This is a constant for rate enquiry.)

Table 15–30 (Cont.) Settings: ProcessSalesOrderShippingChargeLogisticsProvABCSImpl

Property Name	Value/Default Value	Description
Logistics.CalculateShippingCharge.RequestType	LowestCost	OTM Query type is ordered by lower cost.
Logistics.CalculateShippingCharge.WeightUOM	LB	UOM to be used for weight for OTM.
Logistics.CalculateShippingCharge.VolumeUOM	CUFT	UOM to be used for volume for OTM.

Table 15–31shows settings for the InterfaceSyncProductStructureEBF service property:

Table 15–31 Settings: InterfaceSyncProductStructureEBF Service Property

Property Name	Value/Default Value	Description
Routing.ItemCompositionEBS.SyncItemCompositionList.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
Routing.ItemCompositionEBS.SyncItemCompositionList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.ItemCompositionEBS.SyncItemCompositionList.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncrequestrecipient	CAVS Endpoint URI, when CAVS is enabled.
Routing.ItemCompositionEBS.QueryItemCompositionList.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
Routing.ItemCompositionEBS.QueryItemCompositionList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.ItemCompositionEBS.QueryItemCompositionList.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Product.Source.PIP	O2C/MDM Default = O2C	Governs whether the product structure must be synced. Default value is O2C, which synchronizes the product structure.

Table 15–32 shows settings for the RequestProductStructureSiebelReqABCSImpl service property:

Table 15–32 Settings: RequestProductStructureSiebelReqABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	SEBL_01	Siebel system code (such as SEBL_01, defined in OER) from which requests originate for this process.
ABCSExtension.PreXformABMToEBMABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeProcessItemCompositionEBSEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
Routing.ItemCompositionOrchestrationEBS.ProcessItemComposition.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.ItemCompositionOrchestrationEBS.ProcessItemComposition.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
Routing.ItemCompositionOrchestrationEBS.ProcessItemComposition.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponserecipient	CAVS Endpoint URI, when CAVS is enabled.

Table 15–33 shows settings for the CreateProductEbizReqABCSImpl service property:

Table 15–33 Settings: CreateProductEbizReqABCSImpl Service Property

Property Name	Value/Default Value	Description
SourceMileStone	WF_BPEL_Q	Governs the name of the queue on which the consumer would be listening (example: WF_BPEL_Q).
Default.SystemID	EBIZ_01	Ebiz system code (such as EBIZ_01, defined in OER) to which data is synced. The System code value present in the EBM header of the incoming message takes precedence over this property value.
Routing.QuerySimpleItemAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/QuerySimpleItemAdapter/QuerySimpleItemAdapter_ep	Endpoint URI of Ebiz adapter.
Routing.QuerySimpleItemAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.

Table 15–33 (Cont.) Settings: CreateProductEbizReqABCSImpl Service Property

Property Name	Value/Default Value	Description
Routing.ItemEBS.SyncItemList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.ItemEBS.SyncItemList.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
Routing.ItemEBS.SyncItemList.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponserecipient	CAVS Endpoint URI, when CAVS is enabled.
Routing.QuerySimpleItemAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponserecipient	CAVS Endpoint URI, when CAVS is enabled.
BYPASS_ITEMVALIDATIONORG_FLAG	Y/N Default value = N	This property controls whether to bypass the validation of the organization associated with item against OE:Item Validation Org. By default, the organization associated with Item is validated against OE:Item Validation Org.
ABCSExtension.PreXformABMToEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeItemEBSEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
PUBLISH_ALL_PRODUCT	Y/N Default = N	Governs whether to sync both nonorderable and orderable products (when value is Y) or to sync only nonorderable products while performing Item sync (when value is N).

Table 15–34shows settings for the UpdateProductEbizReqABCSImpl service property:

Table 15–34 Settings: UpdateProductEbizReqABCSImpl Service Property

Property Name	Value/Default Value	Description
SourceMileStone	WF_BPEL_Q	Governs the name of the queue on which the consumer would be listening (example WF_BPEL_Q).
Default.SystemID	EBIZ_01	Ebiz system code (such as EBIZ_01, defined in OER) to which data is synced. The system code value present in the EBM header of the incoming message takes precedence over this property value.
Routing.QuerySimpleItemUpdateAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/QuerySimpleItemUpdateAdapter/QuerySimpleItemUpdateAdapter_ep	Endpoint URI of Ebiz adapter.
Routing.QuerySimpleItemUpdateAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
BYPASS_ITEMVALIDATIONORG_FLAG	Y/N Default = N	This property controls whether to bypass the validation of the organization associated with the item against OE:Item Validation Org. By default, the organization associated with the item is validated against OE:Item Validation Org.
ABCSExtension.PreXformABMToEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeItemEBSEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
Routing.ItemEBS.SyncItemList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.ItemEBS.SyncItemList.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
Routing.ItemEBS.SyncItemList.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponserecipient	CAVS Endpoint URI, when CAVS is enabled.
Routing.QuerySimpleItemUpdateAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponserecipient	CAVS Endpoint URI, when CAVS is enabled.

Table 15–35 shows settings for the BulkLoadProductEbizReqABCSImpl service property:

Table 15–35 Settings: BulkLoadProductEbizReqABCSImpl Service Property

Property Name	Value/Default Value	Description
SourceMileStone	WF_BPEL_Q	Governs the name of the queue on which the consumer would be listening (example: WF BPEL_Q).
Routing.TransformAppContext EbizService.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_ PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI when CAVS is enabled.
Routing.TransformAppContext EbizService.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Default.SystemID	EBIZ_01	Ebiz system code (such as EBIZ_01, defined in OER) to which data is synced. The System code value present in the EBM header of the incoming message takes precedence over this property value.
Routing.QuerySimpleItemBulk LoadAdapter.EBIZ_ 01.EndpointURI	http://<SOA_HOST>:<SOA_ PORT>/ soa-infra/services/default/QuerySimpleItemBulkLoadAdapter /QuerySimpleItemBulkLoadAdapter_ep	Endpoint URI of Ebiz adapter.
Routing.QuerySimpleItemBulk LoadAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
BYPASS_ ITEMVALIDATIONORG_FLAG	Y/N Default = N	This property controls whether to bypass the validation of the organization associated with the item against OE:Item Validation Org. By default, the organization associated with the item is validated against OE:Item Validation Org.
ABCSExtension.PreXformABM ToEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeItem EBSEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
Routing.ItemEBS.SyncItemList. RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.

Table 15–35 (Cont.) Settings: BulkLoadProductEbizReqABCSImpl Service Property

Property Name	Value/Default Value	Description
Routing.ItemEBS.SyncItemList.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
Routing.ItemEBS.SyncItemList.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponserecipient	CAVS Endpoint URI, when CAVS is enabled.
Routing.QuerySimpleItemBulkLoadAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponserecipient	CAVS Endpoint URI, when CAVS is enabled.

Table 15–36shows settings for the SyncProductSiebelProvABCSImpl service property:

Table 15–36 Settings: SyncProductSiebelProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	SEBL_01	Siebel system code (such as SEBL_01, defined in OER) from which requests originate for this process.
SIMPLE_PRODUCT_RELEASE_FLAG	Y/N Default = Y	This property controls whether to release Simple Product in Siebel. By default, Simple Product (that is, Bundle product) in Siebel is released.
COMPLEX_PRODUCT_RELEASE_FLAG	Y/N Default = N	This property controls whether to release Complex Product in Siebel. By default, Complex Product (that is, Customizable product) in Siebel is not released.
Routing.SWIProductIntegrationIORes.SEBL_01.EndpointURI	http://<SIEBEL_HOST>/eai_enu/start.swe?SWEExtSource=SecureWebService&SWEExtCmd= Execute&WSSOAP=1	Endpoint URI of Siebel service.
Routing.SWIProductIntegrationIORes.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SWIProductIntegrationIORes.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponserecipient	CAVS Endpoint URI, when CAVS is enabled.
Routing.ItemEBSV2.SyncItemListResponse.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
Routing.ItemResponseEBSV2.SyncItemListResponse.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.

Table 15–36 (Cont.) Settings: SyncProductSiebelProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Routing.ItemResponseEBSV2.SyncItemListResponse.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponserecipient	CAVS Endpoint URI, when CAVS is enabled.
ABCSExtension.PreXformEBMtoABMEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeSWIProductImportUpsertABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostInvokeSWIProductImportUpsertABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostXformABMtoEBMEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
Product.Source.PIP	MDM/O2C Default = O2C	Governs whether O2C or MDM is the product master.

Table 15–37 shows settings for the SyncItemCompositionListSiebelProvABCSImpl service property:

Table 15–37 Settings: SyncItemCompositionListSiebelProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	SEBL_01	Siebel system code (such as SEBL_01, defined in OER) from which requests originate for this process.
Routing.SWIProductIntegration.SEBL_01.EndpointURI	http://<SIEBEL_HOST>/eai_enu/start.swe?SWEExtSource=SecureWebService&SWEExtCmd= Execute&WSSOAP=1	Endpoint URI of Siebel service.
Routing.SWIProductIntegration.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.ItemCompositionEBS.SyncItemCompositionListResponse.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.

Table 15–37 (Cont.) Settings: SyncItemCompositionListSiebelProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Routing.ItemCompositionResponseEBS.SyncItemCompositionListResponse.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.ItemCompositionResponseEBS.SyncItemCompositionListResponse.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>//AIAValidationSystemServlet/asyncrequestrecipient	CAVS Endpoint URI, when CAVS is enabled.
Routing.SWIProductIntegrationIORes.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponserecipient	CAVS Endpoint URI, when CAVS is enabled.
SIMPLE_PRODUCT_RELEASE_FLAG	Y/N Default = Y	This property controls whether to release Simple Product in Siebel. By default, Simple Product (that is, Bundle product) in Siebel is released.
COMPLEX_PRODUCT_RELEASE_FLAG	Y/N Default = N	This property controls whether to release Complex Product in Siebel. By default, Default Complex Product (that is, Customizable product) in Siebel is not released.
ABCSExtension.PreXformEBMtoABMEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeSWIProductImportUpsertABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostInvokeSWIProductImportUpsertABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostXformABMtoEBMEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–38 shows settings for the QueryItemCompositionListEbizProvABCSImpl service property:

Table 15–38 Settings: QueryItemCompositionListEbizProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	EBIZ_01	Ebiz system code (such as EBIZ_01, defined in OER) to which data is synced.
PUBLISH_ALL_PRODUCT	Y/N Default = N	Governs whether the ABCS process should publish all child components. By default, the process publishes only nonorderable child component product.
ABCSExtension.PreXformItemEBMtoABMEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeExplodeItemCompositionEbizAdapterABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostInvokeQueryItemCompositionEbizAdapterABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostXformItemABMtoEBMEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreXformItemStructureEBMtoABMEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreInvokeQueryStructureItemCompositionEbizAdapterABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostInvokeQueryStructureItemCompositionEbizAdapterABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostXformItemStructureABMtoEBMEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
Routing.ExplodeItemCompositionEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.

Table 15–38 (Cont.) Settings: QueryItemCompositionListEbizProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Routing.ExplodeItemCompositionEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/ExplodeItemCompositionEbizAdapter/ExplodeItemCompositionEbizAdapter_ep	Endpoint URI of Ebiz adapter.
Routing.ExplodeItemCompositionEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.QueryItemCompositionEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/QueryItemCompositionEbizAdapter/QueryItemCompositionEbizAdapter_ep	Endpoint URI of Ebiz adapter.
Routing.QueryItemCompositionEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.QueryItemCompositionEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.QueryStructureItemCompositionEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/QueryStructureItemCompositionEbizAdapter/QueryStructureItemCompositionEbizAdapter_ep	Endpoint URI of Ebiz adapter.
Routing.QueryStructureItemCompositionEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.QueryStructureItemCompositionEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.

Table 15–39 shows settings for the CreateAssetSiebelProvABCSImpl service property:

Table 15–39 Settings: CreateAssetSiebelProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Routing.SWIAssetManagementIO.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Default.SystemID	SEBL_01	Siebel system code (such as SEBL_01, defined in OER) from which requests originate for this process.
ABCSExtension.PreProcessABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreProcessEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
Routing.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SWIAssetManagementIO.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SWIAssetManagementIO.SEBL_01.EndpointURI	http://<SIEBEL_HOST>/eai_enu/start.swe?SWEExtSource=SecureWebService&SWEExtCmd=Execute&WSSOAP=1	Endpoint URI of Siebel Service.

Table 15–40 shows settings for the UpdateAssetSiebelProvABCSImpl service property:

Table 15–40 Settings: UpdateAssetSiebelProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	SEBL_01	Siebel system code (such as SEBL_01, defined in OER) from which requests originate for this process.
ABCSExtension.PreProcessABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreProcessEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
Routing.SWIAssetManagementIO.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.

Table 15–40 (Cont.) Settings: UpdateAssetSiebelProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Routing.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SWIAssetManagementIO.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SWIAssetManagementIO.SEBL_01.EndpointURI	http://<SIEBEL_HOST>/eai_enu/start.swe?SWEExtSource=SecureWebService&SWEExtCmd=Execute&WSSOAP=1	Endpoint URI of Siebel Service.

Table 15–41 shows settings for the UpdateItemInstanceEbizReqABCSImpl service property:

Table 15–41 Settings: UpdateItemInstanceEbizReqABCSImpl Service Property

Property Name	Value/Default Value	Description
Routing.QueryItemInstanceEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.QueryItemInstanceEbizAdapter.EBIZ_01.EndpointURI	Ebiz version 11.5.10 EndpointURI should be http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/QueryItemInstanceEbizAdapter/QueryItemInstanceEbizAdapter_ep EBiz version 12.1.x EndpointURI should be http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/QueryItemInstanceEbizR12VersionAdapter/QueryItemInstanceEbizR12VersionAdapter_ep	Endpoint URI of Ebiz adapter.
Routing.QueryItemInstanceOrderEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.

Table 15–41 (Cont.) Settings: UpdateItemInstanceEbizReqABCSImpl Service Property

Property Name	Value/Default Value	Description
Routing.QueryItemInstanceOrderEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/GetItemValidationOrganizationOUEbizAdapter/GetItemValidationOrganizationOUEbizAdapter_ep	Endpoint URI of Ebiz adapter.
Default.SystemID	EBIZ_01	Ebiz system code (such as EBIZ_01, defined in OER) to which data is synced.
ABCSExtension.PreProcessABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreProcessEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
Routing.QueryItemInstanceOrderEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.QueryItemInstanceEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.InstalledProductEBSV2.UpdateInstalledProductList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.InstalledProductEBSV2.UpdateInstalledProductList.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
Routing.InstalledProductEBSV2.UpdateInstalledProductList.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.

Table 15–42 shows settings for the CreateItemInstanceEbizReqABCSImpl service property:

Table 15–42 Settings: CreateItemInstanceEbizReqABCSImpl Service Property

Property Name	Value/Default Value	Description
Routing.QueryItemInstanceEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.QueryItemInstanceEbizAdapter.EBIZ_01.EndpointURI	Ebiz version 11.5.10 EndpointURI should be http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/QueryItemInstanceEbizAdapter/QueryItemInstanceEbizAdapter_ep EBiz version 12.1.x EndpointURI should be http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/QueryItemInstanceEbizR12VersionAdapter/QueryItemInstanceEbizR12VersionAdapter_ep	Endpoint URI of Ebiz adapter.
Routing.GetItemValidationOrganizationOUEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.GetItemValidationOrganizationOUEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/GetItemValidationOrganizationOUEbizAdapter/GetItemValidationOrganizationOUEbizAdapter_ep	Endpoint URI of Ebiz adapter.
Default.SystemID	EBIZ_01	Ebiz system code (such as EBIZ_01, defined in OER) to which data is synced.
ABCSExtension.PreProcessABM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PreProcessEBM	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
Routing.GetItemValidationOrganizationOUEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.QueryItemInstanceEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.

Table 15–42 (Cont.) Settings: CreateItemInstanceEbizReqABCSImpl Service Property

Property Name	Value/Default Value	Description
Routing.InstalledProductEBSV2.CreateInstalledProductList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.InstalledProductEBSV2.CreateInstalledProductList.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes the message to the target service.
Routing.InstalledProductEBSV2.CreateInstalledProductList.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.

Table 15–43 shows settings for the TransformAppContextEbizService service property:

Table 15–43 Settings: TransformAppContextEbizService Service Property

Property Name	Value/Default Value	Description
TransformAppContextEbizService.Soap.EndpointURL	N/A	Endpoint URL for the service.
TransformAppContextEbizService.EBIZ_01.ResponsibilityDVM	ORACLE_RESPONSIBILITY	DVM used to look up the responsibility for the User/OU combination for the instance EBIZ_01.
TransformAppContextEbizService.DefaultUser	OPERATIONS	Default user for initializing Oracle EBS service calls.
TransformAppContextEbizService.DefaultOperatingUnit	204	Default organization for initializing Oracle EBS service calls.
TransformAppContextEbizService.DefaultResponsibility	Order Management Super User, Vision Operations (USA)	Default Oracle EBS responsibility for application context.
Routing.QueryResponsibilityEbizAdapter.RouteToCAVS	false/true	Determines whether the endpoint URL should be routed to either the end application service or CAVS for simulating the service.
Routing.QueryResponsibilityEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	This property sets the endpoint URL for the CAVS simulator.
Default.SystemID	EBIZ_01	The application is responsible for sending the system ID from which the request is being sent. If any requestor application fails to send the system ID, AIA picks the default system ID from this configuration property.
Routing.QueryResponsibilityEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/QueryResponsibilityEbizAdapter	Endpoint URI of the Oracle EBS adapter.

Table 15–44 shows settings for the TransformAppContextSiebelService service property:

Table 15–44 Settings: TransformAppContextSiebelService Service Property

Property Name	Value/Default Value	Description
TransformAppContextSiebelService.Soap.EndpointURL	N/A	Endpoint URL for the service.
TransformAppContextSiebelService.DefaultUser	User1	Default Siebel user.
TransformAppContextSiebelService.DefaultBusinessUnit	SiebelAdmin	Default Siebel business unit.

Table 15–45 shows settings for the SyncSalesOrderEbizProvABCSImpl service property:

Table 15–45 Settings: SyncSalesOrderEbizProvABCSImpl Service Property

Property Name	Value/Default Value	Description
Default.SystemID	EBIZ_01	Ebiz system code (such as EBIZ_01, defined in OER) from which requests originate for this process.
ABCSExtension.PreXformEBMtoABM	true/false; Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
ABCSExtension.PostXformEBMtoABM	true/false; Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
Routing.ProcessSalesOrderEbizAdapter_pttV1.RouteToCAVS	true/false; Default = false	Governs whether the service should route the message to the CAVS endpoint. Default value is false, which does not route the message to CAVS. If set to true, it routes the message to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.ProcessSalesOrderEbizAdapter_pttV1.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.ProcessSalesOrderEbizAdapter_pttV1.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/soa-infra/services/default/ProcessSalesOrderEbizAdapter/ProcessSalesOrderEbizAdapter_ep	Endpoint URI of the Ebiz adapter.
Routing.ProcessSalesOrderEbizAdapter_pttV1.MessageProcessingInstruction.EnvironmentCode	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes to the target service.
ABCSExtension.PreInvokeABS	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.

Table 15–45 (Cont.) Settings: SyncSalesOrderEbizProvABCSImpl Service Property

Property Name	Value/Default Value	Description
ABCSExtension.PostInvokeABS	true/false Default = false	Governs whether ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked.
RMAOrderTypes	RMA	This property defines the value for RMA order types. This is a list of code values that represent RMA order type. Affects the attribute mappings. This is a comma-separated (no spaces) list of string values.
OrderSourceReference	COMMON/SourceSystemIdentifier; Default = COMMON	Property to decide Order Source Reference should hold the Common ID or Source System Identifier.
DefaultOrderSourceId	Default value = 28	Property to supply the default ORDER_SOURCE_ID if dvm lookup fails for quotes.
DefaultQuoteSourceId	Default value = 29	Property to supply the default ORDER_SOURCE_ID if dvm lookup fails for quotes.
OM.CALCULATE_PRICE_FLAG	Default value = P	The CALCULATE_PRICE_FLAG on the order line entity controls whether a pricing or charge calculation should be done on the line. Default is P, which specifies that only the freight charge calculation is done.
HONOR_ORDER_NUMBER	Y/N; Default = N	Governs whether ORDER_NUMBER should be passed as part of inbound ABM. Default value is N which does not pass ORDER_NUMBER.
CreditCardAuthorized	Y/N; Default = Y	Property governing whether to pass credit card details, even when authorization details are missing. Default value is Y, which assumes that authorization details are present.

15.17 Performing Post Installation Configurations

This section includes these post installation configurations for the Order to Cash integration:

- Configuring the Oracle Product Master Data Management pre-built integration with the Order to Cash: Siebel CRM - EBS pre-built integration to enable interoperability
- Deploying services and creating grants to methods
- Subscribe to business events

15.17.1 Configuring the Oracle Product MDM Integration with the Order to Cash: Siebel CRM - EBS Integration

To enable interoperability between the Product MDM integration and the Order to Cash integration, see the *Oracle Application Integration Architecture Pre-Built Integrations: Functional Interoperability Configuration Guide*.

15.17.2 Deploying Services and Creating Grants to Methods

To deploy services and create grants to methods:

1. Add the responsibility for the user (SYSADMIN).
   a. Navigate to **System Administrator, Security: User, Define**.
   b. Query for User (SYSADMIN).
   c. Add responsibility *Integrated SOA Gateway*.
2. Add the role Irep Administrator for the user.
   a. Navigate to **User Management, User**, Search for user, Update, Assign Role.
   b. Search for role Irep Administrator and assign.
3. Deploy service and create grants.
   a. Navigate to Integrated SOA Gateway, Integration Repository, Select View By as Interface Type, Product Life Cycle Management, and Advanced Product Catalog.
   b. Click **Item Service**.
   c. Select the methods and user (who is invoking the service).
4. Create grant.
   a. Select the methods and click **Create Grant**.
   b. Select the user who invokes the service and click **Deploy Service**.

15.17.3 Subscribing to the Business Events

To subscribe to business events:

1. Log in as SYSADMIN/SYSADMIN.
2. Navigate to **Workflow Administrator, Web Applications Responsibility, Business Events**.
3. Query for business event oracle.apps.ego.item.publishItem, or oracle.apps.bom.structure.publishStructure.
4. Click **Subscription**.
5. Enter the details shown in Example 15–2 in the Subscription page:

Example 15–2 Subscription Details

```
System (Select from the LOV),
Source Type -Local,
Event Filter - "oracle.apps.ego.item.publishItem" /
"oracle.apps.bom.structure.publishStructure".
Phase -205
Rule Function-WF_RULE.DEFAULT_RULE,
Out Agent (Select from the LOV),
Owner Name-FND
Owner Tag-FND
```

6. Click **Apply**.

Note: Users can publish only those items on which they have Publish Privilege. By default, this privilege is available with Item Author Role. This privilege can be added to any of the roles. Before publishing, users have to check if they have privilege to publish items/structures.

For more information about installation and related activities, see the *Oracle Application Integration Architecture Installation and Upgrade Guide for Pre-Built Integrations*.

A

Configuring ODI-Based Initial Loads against a Non-Oracle Target Database

This appendix provides instructions as an example of how to set up a data server for a non-Oracle database.

This chapter includes the following section:

- Section A.1, "Configuring ODI-Based Initial Loads Against a Non-Oracle Target Database"

A.1 Configuring ODI-Based Initial Loads Against a Non-Oracle Target Database

Oracle Data Integrator (ODI) is a bulk-loading tool used to load data from one system to another. ODI is used in the Order to Cash: Siebel CRM - EBS pre-built integration for the initial bulk load of assets, customers, products, and price lists using Oracle E-Business Suite (Oracle EBS) as the source system and Siebel Customer Relationship Management (Siebel CRM) as the target system.

ODI provides the flexibility to use source and target systems from a variety of database technologies such as Oracle, MS-SQL, Sybase, DB2, and so on. The code generated for a given source-target combination can be used for any other combination of database systems with little or no modification.

This section describes how to use the ODI maps that are built against an Oracle target database for different database systems. In this section, the Customer bulk load is used as an example. You can follow similar instructions for all of the other initial loads.

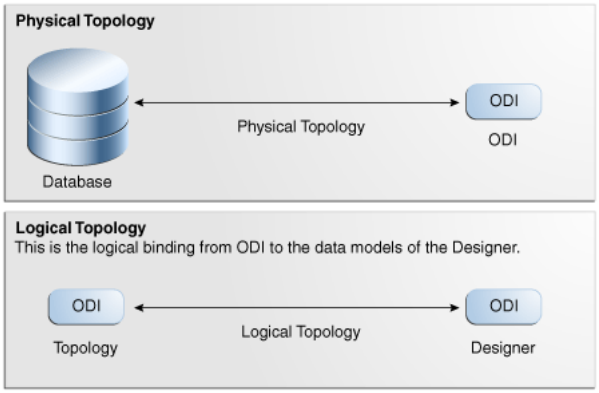
Frequently Used Terms

Here are the definitions of terms used in this section:

- **Topology**: This is a graphical user interface (GUI)-based interface of ODI that makes physical connections to the data objects, such as Database, XML files, and so forth.
  - *Physical Topology:* This defines the connections to the database and file locations for XML files, as shown in Figure A–1.

    This includes DB user, DB password, SID, Host, Port and so forth. You must modify the physical topology to point to the new Source and Target locations, if they are different from what is specified in `deploy.properties`.
  - *Logical Topology:* This is the logical binding from ODI to the data models of the Designer, as shown in Figure A–1.

Figure A–1 Physical and Logical Topology



- **Designer**: This ODI interface defines the data models and the mappings between those data models.

  Data models are logical representations of ODI objects, such as tables, files, views, and so forth.

- **Operator**: This interface displays the run-time status of the bulk loads while they are being carried out.

A.1.1 Overview of Steps

1. Confirm that the bulk load code has been successfully imported into ODI.
2. Delete the Siebel data server from the Topology.

   See Using the Topology Manager, step 4.
3. Create a new Siebel data server under the proper technology.

   See Using the Topology Manager, step 5.
4. Make connections to the database and use the same Logical Schema name and the context that was used previously.

   See Using the Topology Manager, steps 6, 7, and 8.
5. Go to the Model tab in the Designer. Edit the Siebel model and change the technology to the one that is used for Siebel.

   See Using the Designer, step 2.
6. Import the appropriate load and integration knowledge modules, if not present.

   See Using the Designer, step 5.
7. On the Flow tab, make the appropriate changes for the load and integration knowledge modules wherever necessary.

   See Using the Designer, step 6.
8. Save your interfaces and run.

A.1.2 Using the Topology Manager

Follow these steps to set up the database in Topology Manager:

1. Confirm that the bulk load code has been successfully imported into ODI.

2. Log in to the Topology Manager.

   For this example, Siebel is using a database of technology X.

3. Go to the Physical Architecture tab.

   The Technology tab displays a list of technologies for the database, for example: Oracle, MS-SQL, and so on. Expand the Oracle tab and delete the Siebel data server.

4. Right-click the tab and choose the appropriate option from the menu to create a data server in the technology X tab.

5. Enter the appropriate database details (for example, DB username, password, URL, Port, and so forth).

6. Click **Test** to verify your database connection.

   After having successfully connected to the respective database, click **OK** or **Apply**, which takes you to a new window asking for the name of the schema where the Siebel tables are located. These fields are located on the Definition tab. Select the proper schema name from the drop-down menu. Use the same schema name for all references, for example, Work schema.

7. This new window has a Context tab.

   Browse this tab. In the top-left corner of this window, click the **Insert** icon. A new row appears that has two columns: Context and Logical Schema. Select the same context as the project context and give the same logical schema name as it was for the data server under Oracle. Ensure that you give the same logical schema, or else it is not able to connect to Siebel tables and ends in error while running bulk loads. Regarding O2C2, select the context as My_context from the menu and enter Logical schema name SEBL_SRVR_log_sch.

This completes the database setup.

A.1.3 Using the Designer

Follow these steps to use the Designer:

1. Log in to the Designer.

2. Open the Models tab and expand the Siebel model.

   Double-click to edit the Siebel model. Use the drop-down menu to change the technology from Oracle to X. Select the appropriate Siebel schema from the adjacent tab of the Logical Schema, if not selected.

3. Check whether the context from the Reverse tab matches with the context name of project in consideration.

4. Click **OK** to make your changes.

5. Edit the appropriate interfaces using Siebel tables both as Sources and as Targets.

6. Import the LKM SQL to SQL and "IKM SQL to SQL Append knowledge modules into the project.

7. In the interface screen, select the Flow tab.

   Edit the knowledge module for each of the Siebel tables situated both on the Source and on Target side. Change the existing KM to LKM SQL to SQL if the Siebel table is used as the source table and IKM SQL to SQL Append when the Siebel table is on the target side.

8. Save and exit the interface.
9. Regenerate the scenario for command-line execution.

A.1.4 Running the Customer Load: Example

This example is for the Customer initial load. Based on the previous steps, you can perform the steps as follows:

1. Log in to the Designer.
2. Open the Models tab.

   You should see this Model folder: OracleEbiz_To_Siebel_Model. Expand this folder. You should see the following models:

   - AIA Configuration Model
   - AIA Country Model
   - AIA Gender Model
   - AIA Initials Model
   - AIA State Model
   - OracleEbiz Model
   - Siebel Model
   - Xref Model
3. Double-click the Siebel Model to edit it.

   In the Definition tab of the Model window, change the technology from Oracle to X.
4. Ensure that you have selected SEBL_SRVR_log_sch as the logical schema in the Logical Schema tab.

   Click **OK** and ignore all the errors.
5. Open the Projects tab and expand the Project of the name CustomerInitialBulkLoad.

   Expand the folder Customer_Bulk_Load. Expand the Interfaces tab. The listed interfaces include:

   - 14_Ebiz_Cust_Eim_Account_Interface
   - 15_Ebiz_Cust_Eim_Contact_Interface
   - 16_Ebiz_Cust_Eim_Addr_Per_Interface
   - 17_Ebiz_Cust_Eim_Fn_Accnt1_Interfac
   - 18_Siebel_CUSTOMERPARTY_PARTYID_XRF
   - 19_Siebel_CUSTOMERPARTY_ACCOUNTID_X
   - 20_Siebel_CUSTOMERPARTY_ADDRESSID_X
   - 21_Sbl_CUSTMERPARTY_PARTYLOCATIONID
   - 22_Sbl_CUSTOMERPARTY_LOCATIONREFID
   - 23_Sbl_CUSTOMERPARTY_PARTYCONTACTID
   - 24_Sbl_CUSTOMERPARYTY_CONTACTID

- 25_Sbl_CUSTOMERPARTY_CONTACT_COMMID
- 26_Sbl_CUSTOMERPARTY_ACCOUNT_COMMID

6. Expand the Knowledge Module under Customer_Bulk_Load and right-click the Loading (LKM) option.

   Click the **Import Knowledge Modules** option. This opens a window that displays the current directory and knowledge modules present in that directory. Change the directory, if required. Select LKM SQL to SQL. Similarly, right-click the **Integration** (IKM) option and select IKM SQL to SQL Append.

7. After importing the Knowledge Modules, open each of the previously mentioned interfaces (from 14 to 26) one after the other and select the Flow tab in each of the Interfaces.

   a. For interfaces 14, 15, and 16, click Target+Staging Area.

      You see a window in the lower half, displaying the IKM used. Change the IKM to IKM SQL to SQL Append and save. In the same window, set FLOW_CONTROL and STATIC_CONTROL to NO.

   b. For interface 17, click EIM_ADDR_PER and select LKM SQL to SQL.

      Click Target+Staging Area. Change the IKM to IKM SQL to SQL Append and save. In the same window, set FLOW_CONTROL and STATIC_CONTROL to NO.

   c. For the rest of the interfaces, (18 to 26), in the Flow tab, select each box and change LKM to LKM SQL to SQL and IKM to IKM SQL to SQL Append (STATIC_CONTROL=NO and FLOW_CONTROL=NO).

   d. Save each interface after making changes in LKM and IKM.

8. Expand PACKAGE under Customer_Bulk_Load. The packages include:

   - OracleEbiz_to_XREF_to_EIM
   - Siebel_to_XREF

9. Expand: Package -> OracleEbiz_to_XREF_to_EIM -> Scenarios.

   Right-click the scenario ORACLEEBIZ_TO_XREF_TO_EIM and select Regenerate from the menu.

10. Expand: Package -> Siebel_to_XREF -> Scenarios.

    Right-click the scenario SIEBEL_TO_XREF and select Regenerate from the menu.

11. You can now run the packages OracleEbiz_to_XREF_to_EIM and Siebel_to_XREF.

> **Note:** You can also run the packages from the command line.

B

Organization Data Setup for Product Synchronization

For the product synchronization from Oracle E-Business Suite (Oracle EBS) to Siebel Customer Relationship Management (Siebel CRM) to execute as designed, certain entities must be established appropriately in the participating applications. This appendix discusses organization definitions and relationships in participating applications, Oracle Application Integration Architecture (Oracle AIA) mapping, and Order to Cash product synchronization behavior.

As part of the business flow of the Order to Cash: Siebel CRM - EBS pre-built integration, the primary objective of synchronizing items from Oracle EBS to products in Siebel CRM is to enable a customer service representative (CSR) to place an order in Siebel CRM for these products and for the Order flow to submit this order in Oracle EBS.

This appendix contains the following sections:

B.1 Organization Definitions and Relationships in the Participating Applications

This section discusses the definitions and relationships of organizations in the following participating applications:

- Oracle EBS
- Siebel CRM

B.1.1 Oracle EBS

This section discusses these definitions and relationships of the organizations in Oracle EBS:

- Operating Units
- Inventory Organizations
- Item Master Organization
- Item Validation Organization

Operating Units: An operating unit is a logical organization within a company that the company management decides to operate. Order transactions are owned by the operating unit organizations.

The transactions for an operating unit are restricted to using the reference data for that same operating unit. That is, all the sales orders (transactional entities) are not only owned by the operating unit on the transaction side, but the reference data is also owned (namely, customer accounts) or associated (namely, items).

Inventory Organizations: Inventory organizations represent manufacturing and storage facilities. Each inventory organization belongs to one parent operating unit. Oracle implements storage facilities, warehouses, and distribution centers in inventory organizations.

Item Master Organization: There can be many inventory organizations in Oracle EBS and one of them is identified with the role of *Item Master Organization*.

> **Note:** It is a best practice in an Oracle EBS implementation to have only one master organization, even though technically it may be possible to have multiple master organizations.

Items are the first to be set up and defined in this item master organization. An item master organization holds a single definition of items that can be shared across many inventory organizations.

The unique key to identify an item in Oracle EBS is Inventory Item Id and Inventory Organization.

Item Validation Organization: In the Order Management module, there is a system parameter called Item Validation Organization defined at the operating-level. This is the inventory organization used by Order Management to validate items when orders are placed against that operating unit. Therefore, when the items are defined, they must be associated to this inventory organization. The OE: Item Validation Organization is also called Item Validation Org.

- Because the master organization is an inventory organization, the master organization might also be identified as the item validation org for an operating unit.
- The same inventory organization can be the item validation org. for multiple operating units.
- The Operating Unit to Item Validation Org. relationship is different from Parent Operating Unit to Inventory Organization relationship. Although in certain implementations, the customers may choose to designate an inventory organization as the item validation org. for an operating unit (OU), where this operating unit also happens to be the parent operating unit for this inventory organization.
- Only the items that are associated with the inventory organization that are also established as item validation org. are visible in the respective operating unit when an order is placed in the same.

B.1.2 Siebel CRM

This section discusses these definitions and relationships of the organizations in Siebel CRM:

- Business Units

- Inventory Locations

Business Units: The business unit organization in Siebel CRM allows the implementation company to partition itself into logical groups. This allows the information associated with the business unit to be visible only to the end-users associated to that business unit (BU).

The transaction data in Siebel CRM (that is, Sales Order), is always associated to the primary business unit. In Siebel CRM, although an order is associated to a specific business unit, products from different business units can be associated on the order lines. In other words, unlike Oracle EBS, the reference data for a transaction can belong to a different organization in Siebel CRM.

A product in Siebel CRM is always associated with a business unit, which becomes its primary business unit. Multiple business units can also be associated with a product.

The unique key to a product in Siebel CRM is Product Name, Business Unit, and Vendor Account (optional - not mapped).

Inventory Locations: Inventory locations in Siebel CRM are used to identify where products are stored, and the source from which the product is fulfilled. An inventory location may be a warehouse, a field office, or it may be virtual location. An inventory location is also associated to a business unit.

Note: In Siebel CRM there is no equivalent of the Oracle EBS item master organization or item validation organization.

B.2 Oracle AIA Mapping

This section discusses:

- Organization mapping
- Inventory location mapping

B.2.1 Organization Mapping

The concepts of business units in Siebel CRM, and operating units in Oracle EBS map directly. Therefore, the existence of one-to-one mapping for them across the applications in the Order to Cash: Siebel CRM - EBS pre-built integration is assumed. See Table B–1 for an example.

For more information about how to set up an organization cross-reference mapping, see Section B.3, "Order to Cash Product Synchronization Behavior."

Table B–1 Organization Mapping Example

Siebel CRM Business Unit	Oracle EBS Operating Unit	Oracle AIA
S_BU1	O_OU_1	AIA_1
S_BU2	O_OU_2	AIA_2
S_BU3	O_OU_3	AIA_3

Many operating units can be defined in Oracle EBS, or multiple business units defined in Siebel CRM. However, from an order processing perspective, only specific business units and operating units must be mapped in the cross-reference where order capture is performed in Siebel CRM and the corresponding fulfillment in Oracle EBS.

B.2.2 Inventory Mapping

There can be multiple inventory organizations in Oracle EBS for an item. However, from an order capture perspective, only those inventory organizations that are associated as item validation organizations for the item in Oracle EBS should be defined as inventory locations in Siebel CRM.

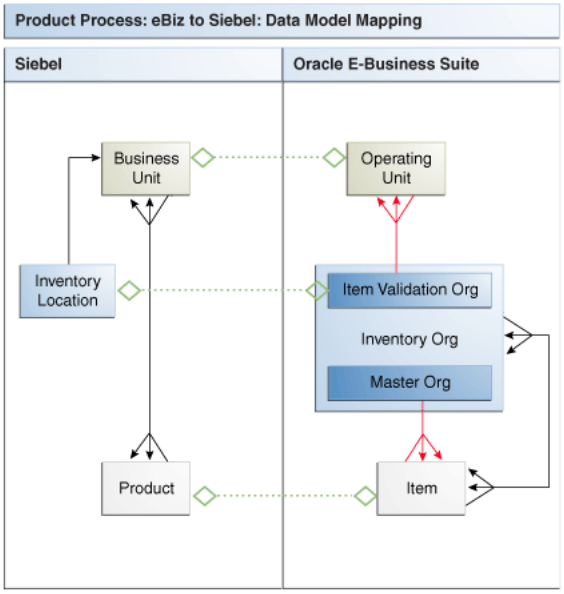
The business unit of the inventory location being created in Siebel CRM should be mapped to the operating unit of the item validation org. in Oracle EBS.

Note: The INVENTORY_LOCATION_ID XREF has an additional attribute SEBL_01_INVLOC_BU, to capture the primary business unit in Siebel CRM for a given inventory location. It is optional, and must be set up only if the item validation organization is shared by multiple operating units. In such cases, the primary business unit for the inventory location must be defined.

For more information about this use case, see Order to Cash Product Synchronization Behavior, Section B.3.2, "Scenario #2: The Item Validation Org. is Shared Across Multiple Operating Units" and Section B.3.3, "Scenario #3: The Item Master Organization is the Item Validation Org. and Shared Across Multiple Operating Units."

Figure B–1 illustrates the logical data mapping for Oracle EBS to Siebel CRM:

Figure B–1 Product Integration: Oracle EBS to Siebel CRM Logical Data Map



B.3 Order to Cash Product Synchronization Behavior

This section discusses cases when an item in Oracle EBS is synchronized as a product in Siebel CRM. Also included are the following best practice scenarios about how to set up the organization cross-reference mappings:

- Section B.3.1, "Scenario #1: Each Operating Unit has a Distinct (Non-Shared) Item Validation Org."

- Section B.3.2, "Scenario #2: The Item Validation Org. is Shared Across Multiple Operating Units"
- Section B.3.3, "Scenario #3: The Item Master Organization is the Item Validation Org. and Shared Across Multiple Operating Units"

An item from Oracle EBS is synchronized as a product in Siebel CRM:

- For every item Id and inventory organization combination in Oracle EBS only where the inventory organization also happens to be an item validation organization. Therefore, not all combinations of item Id and inventory organization Id from Oracle EBS creates a unique product in Siebel CRM.
- The Siebel CRM product's business unit corresponds to the Oracle EBS operating unit that has the inventory organization as the item validation organization.

  If in Oracle EBS, the item is associated to multiple operating units by distinct (non-shared) item validation orgs, then the item records associated with the item validation orgs in all these multiple operating units are synchronized, and cross-references created. Therefore, in Siebel CRM the same product is created for different business units individually.
- Because the item validation org. can be shared across multiple operating units in Oracle EBS, all of the corresponding business units may have to be associated to the product in Siebel CRM. This depends on how product visibility is set up in Siebel CRM. If catalog-based (enterprise-level) visibility is set up in Siebel CRM, then associating additional business units may not be required. However, if organization-based visibility is set up in Siebel CRM for products, then the additional non-primary (multi-org) business unit must be manually associated to the product.
- If the master organization is also the item validation org, then those item records with the master organization Ids are synchronized, and the cross-reference created. If the master organization is not the item validation org, then the item records with the master organization are ignored for creating the cross-reference.

Based in the previous information, there can be various combinations of organization setup and Item and product definition across the two applications. This appendix provides a few best practice scenarios about how to set up the organization cross-reference mappings, and based on this setup how product synchronization to Siebel CRM takes place. How the product cross-reference is established is also described.

Note: There may be other setup options that you make that may not be supported and therefore, may require customization.

B.3.1 Scenario #1: Each Operating Unit has a Distinct (Non-Shared) Item Validation Org.

The scenario description is:

- Only one master organization exists in Oracle EBS. All items are defined in the master organization.
- Each operating unit is associated with a distinct item validation org. In other words, item validation org. is not shared across operating units.
- The item master organization is not associated as the item validation org. to any of the operating units in which an order can be placed.

- The parent operating unit of the inventory organization may or may not be an operating unit in which orders can be placed.

Table B–2 lists the operating units (OU) in Oracle EBS:

Table B–2 Operating Units (OU) in Oracle EBS

#	Name	ID	Order Capture OU	OE Item Validation Org
1	United States OU	504	No	na
2	Purple OU	505	Yes	Beta IO
3	Red OU	506	Yes	Gamma IO
4	Yellow OU	507	Yes	Delta IO
5	White OU	509	No	na
6	Orange OU	510	No	na
7	Green OU	511	No	na
6	Blue OU	512	No	na

Table B–3 lists the organizations (IO) in Oracle EBS:

Table B–3 Organizations (IO) in Oracle EBS

#	Name	ID	Parent Operating Unit	Item Master Org	Does IO serve as Item Validation Org
1	Alpha IO (Item Master Org)	750	United States OU	Yes	No
2	Beta IO	751	White OU	No	Yes
3	Gamma IO	752	Orange OU	No	Yes
4	Delta IO	753	Green OU	No	Yes
5	Phi IO	754	Blue OU	No	No
6	Kappa IO	755	Purple OU	No	No

Table B–4 lists business units (BU) in Siebel

Table B–4 Business Units (BU) in Siebel

#	Name	ID	Used for Order Capture
1	Purple BU	1-AC	Yes
2	Red BU	1-AD	Yes
3	Yellow BU	1-AE	Yes
4	Brown BU	1-AG	No

Table B–5 lists inventory locations (IL) in Siebel. (These are item validation orgs. from Oracle EBS).

Table B–5 Inventory Locations (IL) in Siebel

#	Name	ID	Primary Business Unit
1	Beta IL	1-XB	Purple BU
2	Gamma IL	1-XC	Red BU

Table B–5 (Cont.) Inventory Locations (IL) in Siebel

#	Name	ID	Primary Business Unit
3	Delta IL	1-XC	Yellow BU

Table B–6 lists AIA_ORGANIZATION_ID X-Ref

Table B–6 AIA_ORGANIZATION_ID X-Ref

#	Oracle EBS ID	Common	Siebel ID
1	505 (Purple OU)	C1	1-AC (Purple BU)
2	506 (Red OU)	C2	1-AD (Red BU)
3	507 (Yellow OU)	C3	1-AE (Yellow BU)

Note: The values in parenthesis are just for explanation, they do not go into the X-Ref.

Table B–7 lists AIA_INVENTORY_ID X-Ref

Table B–7 AIA_INVENTORY_ID X-Ref

Oracle EBS ID	Common	Siebel ID
751 (Beta IO)	C51	1-XB (Beta IL)
752 (Gamma IO)	C52	1-XC (Gamma IL)
753 (Delta IO)	C53	1-XD (Delta IL)

Note: The values in parenthesis are just for explanation, they do not go into the X-Ref.

Based on the organization setup, the following tables illustrate how the Oracle EBS item is synchronized to a Siebel product.

Table B–8 lists the item definition in Oracle EBS:

Table B–8 Item Definition in Oracle EBS

#	Item Name	ID	Inventory Org. ID	Parent OU
1	Item A	10	750 (Alpha IO)	504 (US OU)
2	Item A	10	751 (Beta IO)	509 (White OU)
3	Item A	10	752 (Gamma IO)	510 (Orange OU)
4	Item A	10	753 (Phi IO)	512 (Blue OU)
5	Item B	20	751 (Beta IO)	509 (White OU)
6	Item B	20	755 (Kappa IO)	505 (Purple OU)

Table B–9 lists the product definition in Siebel.

Table B–9 Product Definition in Siebel

#	Product Name	Row	Primary Bus. Unit ID	Secondary Bus. Unit ID
1	Item A	1-PD1	1-AC (Purple BU)	na
2	Item A	1-PD2	1-AD (Red BU)	na
3	Item B	1-PD3	1-AC (Purple BU)	na

The Oracle AIA X-Ref contains the information listed in Table B–10 for this product synchronization scenario.

Table B–10 Oracle AIA X-Ref

#	Type	Siebel ID	Oracle EBS ID	Common
1	ITEM_ITEMID	1-PD1	10::751::509	C101
2	ITEM_ITEMID	1-PD2	10::752::510	C102
3	ITEM_ITEMID	1-PD3	20::751::509	C103

B.3.2 Scenario #2: The Item Validation Org. is Shared Across Multiple Operating Units

The scenario description is:

- Only one master organization in Oracle EBS. All items are defined in this master organization.
- The same inventory organization is specified as the item validation organization in the profile option across multiple order capturing operating units.
  - This implies that there is a single corresponding inventory location created in Siebel CRM. However, the business unit to be specified for this single inventory location in Siebel CRM may be corresponding to any of the operating units (for that item validation org.).
  - Whichever business unit is chosen to be specified for the inventory location becomes the primary business unit on the product created during synchronization.
  - All other business units for the given product must be manually associated within the Siebel product definition. This may only be necessary if organization-based visibility is set up in Siebel CRM for the users. (If default catalog-based visibility is used, manually associating the other business units may not be required.)
- The master organization may be this item validation org. across multiple operating units in which an order can be placed.
- The parent operating unit associated with the inventory organization may or may not be the operating unit against which orders are going to be placed.

Table B–11 lists the operating units (OU) in Oracle EBS:

Table B–11 Operating Units (OU) in Oracle EBS

#	Name	ID	Order Capture Operating Unit	OE Item Validation Org
1	United Stated OU	504	No	na
2	Purple OU	505	Yes	Beta IO
3	Red OU	506	Yes	Beta IO

Table B–11 (Cont.) Operating Units (OU) in Oracle EBS

#	Name	ID	Order Capture Operating Unit	OE Item Validation Org
4	Yellow OU	507	Yes	Beta IO
5	White OU	509	No	na
6	Orange OU	510	No	na
7	Green OU	511	No	na
8	Blue OU	512	No	na

Table B–12 lists the inventory organizations (IO) in Oracle EBS:

Table B–12 Inventory Organizations (IO) in Oracle EBS

#	Name	ID	Parent Operating Unit	Item Master Organization	Does IO serve as Item Validation Org.
1	Alpha IO (Item Master Organization	750	United States OU	Yes	No
2	Beta IO	751	White OU	No	Yes
3	Gamma IO	752	Orange OU	No	No
4	Delta IO	753	Green OU	No	No
5	Phi IO	754	Blue OU	No	No
6	Kappa IO	755	Purple OU	No	No

Table B–13 lists business units (BU) in Siebel

Table B–13 Business Units (BU) in Siebel

#	Name	Row	Used for Order Capture
1	Purple BU	1-AC	Yes
2	Red BU	1-AD	Yes
3	Yellow BU	1-AE	Yes
4	Brown BU	1-AG	No

Table B–14 lists inventory locations (IL) in Siebel. (This is the item validation org. from Oracle EBS.)

Table B–14 Inventory Locations (IL) in Siebel

#	Name	ID	Primary Business Unit
1	Beta IL	1-XB	Purple BU

Table B–15 lists the AIA_ORGANIZATION_ID X-Ref

Table B–15 AIA_ORGANIZATION_ID X-Ref

#	Oracle EBS ID	Common	Siebel ID
1	505 (Purple OU)	C1	1-AC (Purple BU)
2	506 (Red OU)	C2	1-AD (Red BU)

Table B–15 (Cont.) AIA_ORGANIZATION_ID X-Ref

#	Oracle EBS ID	Common	Siebel ID
3	507 (Yellow OU)	C3	1-AE (Yellow BU)

Note: The values in parenthesis are just for explanation, they do not go into the X-Ref.

Table B–16 lists the AIA_INVENTORY_LOCATION_ID X-Ref

Table B–16 AIA_INVENTORY_LOCATION_ID X-Ref

#	Oracle EBS ID	Common	SEBL_01	SEBL01_INVLOC_BU
1	751 (Beta IO)	C51	1-XB (Beta IL)	1-AC (Purple BU)

Note: The values in parenthesis are just for explanation, they do not go into the X-Ref.

Based on the organization setup, the following tables illustrate how the Oracle EBS item is synchronized as a Siebel product.

Table B–17 lists the item definitions in Oracle EBS:

Table B–17 Item Definitions in Oracle EBS

#	Item Name	ID	Inventory Org. ID	Parent Operating Unit
1	Item A	10	750 (Alpha IO)	504 (US OU)
2	Item A	10	751 (Beta IO)	509 (White OU)
3	Item A	10	752 (Gamma IO)	510 (Orange OU)
4	Item A	10	753 (Phi IO)	512 (Blue OU)
5	Item B	20	750 (Alpha IO)	504 (US OU)
6	Item B	20	751 (Beta IO)	509 (White OU)
7	Item B	20	755 (Kappa IO)	505 (Purple OU)

Table B–18 lists the product definitions in Siebel

Table B–18 Product Definitions in Siebel

#	Product Name	Row ID	Primary Bus. Unit ID	Non-Primary Bus. Unit ID (Optional)
1	Item A	1-PD1	1-AC Purple BU)	1-AD (Red BU 1-AE (Yellow BU)
2	Item B	1-PD3	1-AC (Purple BU)	1-AD (Red BU 1-AE (Yellow BU)

Note: If required, the non-primary business unit (multi-org) must be manually set up for the product.

The Oracle AIA X-Ref contains the information listed in Table B–19 for this product synchronization scenario.

Table B–19 Oracle AIA X-Ref

#	Type	Siebel ID	Oracle EBS ID	Common
1	ITEM_ITEMID	1-PD1	10::751::509	C101
2	ITEM_ITEMID	1-PD3	20::751::509	C103

B.3.3 Scenario #3: The Item Master Organization is the Item Validation Org. and Shared Across Multiple Operating Units

The scenario description is:

- There is only one item master organization in Oracle EBS. All items are defined in this master organization.
- The item master organization also serves as the item validation organization in the profile option across multiple order capturing operating units.
  - This implies that there is a single corresponding inventory location created in Siebel CRM. However, the business unit to be specified for this single inventory location in Siebel CRM may be corresponding to any of the operating units (for that item validation org.).
  - Whichever business unit is chosen to be specified for the inventory location becomes the primary business unit on the product created during synchronization.
  - All other related business units for the given product must be manually associated within the Siebel product definition. This may only be necessary if organization-based visibility is set up in Siebel CRM for the users. (If default catalog-based visibility is used, manually associating the other business units may not be required.)
- The parent operating unit associated with the inventory organization may or may not be the operating unit against which orders are going to be placed.

Table B–20 lists the operating units (OU) in Oracle EBS:

Table B–20 Operating Units (OU) in Oracle EBS

#	Name	ID	Order Capture Operating Unit	OE Item Validation Org.
1	United States OU	504	No	na
2	Purple OU	505	Yes	Alpha IO
3	Red OU	506	Yes	Alpha IO
4	Yellow OU	507	Yes	Alpha IO
5	White OU	509	No	na
6	Orange OU	510	No	na
7	Green OU	511	No	na
8	Blue OU	12	No	na

Table B–21 lists the inventory organizations (IO) in Oracle EBS:

Table B–21 Inventory Organizations (IO) in Oracle EBS

#	Name	ID	Parent Operating Unit	Item Master Org.	Does IO serve as item Validation Org
1	Alpha IO (Item Master organization)	750	United States OU	Yes	Yes
2	Beta IO	751	White OU	No	No
3	Gamma IO	752	Orange OU	No	No
4	Delta IO	753	Green OU	No	No
5	Phi IO	754	Blue OU	No	No
6	Kappa IO	755	Purple OU	No	No

Table B–22 lists business units (BU) in Siebel

Table B–22 Business Units (BU) in Siebel

#	Name	Row ID	Used for Order Capture
1	Yes	Purple BU	1-AC
2	Yes	Red BU	1-AD
3	Yes	Yellow BU	1-AE
4	No	Brown BU	1-AG

Table B–23 lists inventory locations (IL) in Siebel. (This is the item validation org. from Oracle EBS.)

Table B–23 Inventory Locations (IL) in Siebel

#	Name	ID	Primary Business Unit
1	Alpha IL	1-XM	Purple BU

Table B–24 lists AIA_ORGANIZATION_ID X-Ref

Table B–24 AIA_ORGANIZATION_ID X-Ref

#	Oracle EBS ID	Common	Siebel ID
1	505 (Purple OU)	C1	1-AC (Purple BU)
2	506 (Red OU)	C2	1-AD (Red BU)
3	507 (Yellow OU)	C3	1-AE (Yellow BU)

Note: The values in parenthesis are just for explanation, they do not go into the X-Ref.

Table B–25 lists the AIA_INVENTORY_LOCATION_ID X-Ref

Table B–25 AIA_INVENTORY_LOCATION_ID X-Ref

#	Oracle EBS ID	Common	SEBL_01	SEBL01_INVLOC_BU
1	750 (Alpha IO)	C51	1-XM (Alpha IL)	1-AC (Purple BU)

Note: The values in parenthesis are just for explanation, they do not go into the X-Ref.

Based on the organization setup, the following tables illustrate how the Oracle EBS item is synchronized to a Siebel product.

Table B–26 lists the item definitions in Oracle EBS:

Table B–26 Item Definitions in Oracle EBS

#	Item Name	ID	Inventory Org. ID	Parent Operating Unit
1	Item A	10	750 (Alpha IO)	504 (US OU)
2	Item A	10	751 (Beta IO)	509 (White OU)
3	Item A	10	752 (Gamma IO)	510 (Orange OU)
4	Item A	10	753 (Phi IO)	512 (Blue OU)
5	Item B	20	750 (Alpha IO)	504 (US OU)
6	Item B	20	751 (Beta IO)	509 (White OU)
7	Item B	20	755 (Kappa IO)	505 (Purple OU)

Table B–27 lists the product definitions in Siebel

Table B–27 Product Definitions in Siebel

#	Product Name	Row ID	Primary Bus. Unit ID	Non-Primary Bus. Unit ID (Optional)
1	Item A	1-PD1	1-AC Purple BU)	1-AD (Red BU 1-AE (Yellow BU)
2	Item B	1-PD3	1-AC (Purple BU)	1-AD (Red BU 1-AE (Yellow BU)

Note: If required, the non-primary business unit (multi-org) must be manually set up for the product.

The Oracle AIA X-Ref contains the information listed in Table B–28 for this product synchronization scenario.

Table B–28 Oracle AIA X-Ref

#	Type	Siebel ID	Oracle EBS ID	Common
1	ITEM_ITEMID	1-PD1	10::750::504	C101
2	ITEM_ITEMID	1-PD3	20::750::504	C103

C

Maintaining the Same Order Number Across Siebel and Oracle E-Business Suite

In both the Siebel Order Management and the Oracle E-Business Suite (Oracle EBS) Order Management applications, the order number on the Sales Order is automatically generated. Therefore, the value of the Siebel order number is different from the Oracle EBS order number. By default, the order process integration in this solution allows implementers to maintain the respective order numbers in the two applications.

However, some implementations in their order integration business process may want to carry the order number generated by the Siebel Order Management application into the Oracle EBS Order Management application. This appendix provides the details on how to maintain the same order number across the two applications.

This appendix includes the following sections:

C.1 Setting Up Oracle EBS

To set up Oracle EBS:

1. Login to Oracle EBS, and set up the new Transaction Type assigned to Document Sequence with manual numbering.
2. Navigate to **Setup, Transaction Types, Define** and query for the transaction type created in Step 1.
3. Navigate to **Help, Diagnostics, Examine** and enter *Transaction_Type_Id* in the **LOV** field.
4. Click **Tab**.
5. Make a note of the value.

   This value is used to setup the ORDER_TYPE domain value map (DVM) in Oracle Application Integration Architecture (Oracle AIA).

Caution: This change is recommended for only new Order Types in Oracle EBS, and not to existing ones.

C.2 Setting Up Oracle AIA DVMs

The various order types for Oracle EBS are managed using the Oracle AIA *ORDER_TYPE* DVM.

To edit the value of the ORDER_TYPE DVM:

1. In the DVM file <AIAHOME>/AIAMetaData/dvm/ORDER_TYPE.dvm, select the record where value of the **COMMON** column is *SALESORDER*.
2. Change the value of column **EBIZ_01** from *1437* to *Transaction_Type_Id* (as recorded from "Setting Up Oracle EBS").
3. Update MDS. Modify <AIAHOME>/aia_instances/MDM/config/UpdateMetaDataDP.xml to include the file modified in step 1. Then navigate to the folder <AIAHOME>/Infrastructure/Install/config and run:

   ```
   ant -f UpdateMetaData.xml-

   DPropertiesFile<AIAHOME>/aia_instances/O2C/config/AIAInstallProperties.xml
   ```

4. After completion, login to the Oracle AIA console, navigate to setup\AIA Configuration and click **Reload** to load the changes.

C.3 Configuring AIAConfigurationsProperties.xml

When a sales order is submitted in Siebel CRM and sent for processing, the **Order Number** field is included in the message. The sales order submitted to Oracle EBS may contain the order number provided by Siebel CRM. In the Oracle AIA Order Provider Service the HONOR_ORDER_NUMBER property controls this behavior.

The AIAconfigurationProperties.xml file includes the service-level configuration property HONOR_ORDER_NUMBER (values: *Y* or *N*, default is *N*).

To maintain the same order number across the Siebel and Oracle EBS applications:

- In the AIAConfigurationsProperties.xml file, change the value of the HONOR_ORDER_NUMBER property to *Y*.

C.4 Solution Assumptions and Constraints

1. This change is recommended for only new Order Types in Oracle EBS, and not to existing Order Types.
2. If you decide to implement this feature, it is mandatory to configure Siebel such that the Siebel Order Number field must contain only numeric values.

   This is because, by default, the Order Number field in Siebel can contain alphanumeric characters. However, the Order Number attribute in Oracle EBS can store only numeric values.

D

Composite Application Validation System Changes

This appendix discusses how the Composite Application Validation System (CAVS) has changed from the Oracle Application Integration Architecture (Oracle AIA) Communications 11.1 release to the Oracle AIA Communications 11.2 release and provides details on Requestor application business connector services (ABCSs) and Provider ABCSs.

This appendix includes the following sections:

- Section D.1, "Configuration Properties for CAVS Enablement in 11.1"
- Section D.2, "Configuration Properties for CAVS Enablement in 11.2"

D.1 Configuration Properties for CAVS Enablement in 11.1

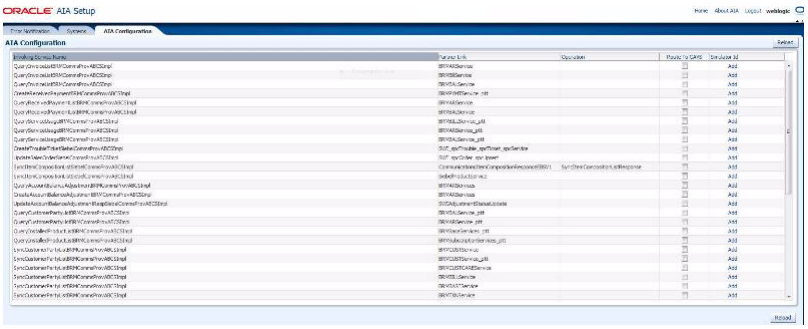
In the 11.1 Oracle AIA CAVS implementation, every service has a number of configuration properties.

For more information about these configuration properties, see the *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*.

For the `RouteToCAVS` property, the out-of-the-box value is *False*. Oracle AIA provides a user interface (UI), which allows the user to toggle this property value between *True* and *False* for each service listed.

Figure D–1 is an example of the AIA Configuration UI. To navigate to this UI: log in to the AIA Console (http://<host>:<port>/AIA), go to **Setup** and then select the **AIA Configuration** tab.

Figure D–1 Oracle AIA Configuration Screen



D.2 Configuration Properties for CAVS Enablement in 11.2

CAVS enablement has been reorganized. As a result, the UI can no longer be used to toggle the value of the `RouteToCAVS` property for the Order to Cash services.

The following instructions describe how to modify the configuration properties for Requestor ABCS and Provider ABCS to enable CAVS.

> **Note:** Any change in the System Configuration screen does not enable CAVS for a service. You must make changes manually in the Oracle AIA configuration file to make the service CAVS enabled.

D.2.1 Requestor ABCS

For CAVS enablement of Requestor ABCS, a single configuration property is maintained.

For example,
`EBSOverride.SalesOrderEBSV2.ProcessSalesOrderFulfillment.Address`

In order to enable CAVS, you must manually edit the AIAConfigurationProperties.xml file, which is located here: $AIA_HOME/aia_instances/$INSTANCE_NAME/AIAMetaData/config. Entries in the AIAConfigurationProperties.xml file are case sensitive.

To enable CAVS for Requestor ABCS:

1. Open the AIA configuration properties file.
2. Set the **Address** property to the CAVS URI for each service that you want to be CAVS enabled.

   For example, http://<soa_server>:<soa_port>/AIAValidationSystemServlet/asyncrequestrecipient
3. Save and close the file after you have set this property for all desired Requestor ABCSs.
4. Login to the AIA Console (http://<host>:<port>/AIA). Go to **Setup**, and then select the **AIA Configuration** tab. Click **Reload** to relaod the configuration file and make your changes effective.

D.2.2 Provider ABCS

For CAVS enablement of a Provider ABCS, two configuration properties are maintained. For example:

- ""Routing.SWI_spcOrder_spcUpsert.RouteToCAVS"
- ""Routing.SWI_spcOrder_spcUpsert.SEBL_01.EndpointURI"

In order to enable CAVS, you must manually edit the AIAConfigurationProperties.xml file, which is located here: $AIA_HOME/aia_instances/$INSTANCE_NAME/AIAMetaData/config. Entries in the AIAConfigurationProperties.xml file are case sensitive.

To enable CAVS for Provider ABCS:

1. Open the AIA configuration properties file.
2. Set the `RouteToCAVS` property value to *True* and set the EndpointURI property value to the actual CAVS URL for each service that you want to be CAVS enabled.

3. Save and close the file after you have set this property for all desired Provider ABCSs.
4. Login to the AIA Console (http://<host>:<port>/AIA). Go to **Setup**, and then select the **AIA Configuration** tab. Click **Reload** to relaod the configuration file and make your changes effective.

E

Reintroducing Enterprise Business Services

Enterprise business services (EBSs) are used to help route to multiple Providers. If you are using one source and one target system for your integration flows then EBSs are unnecessary. However, if you must dynamically identify a Provider system during runtime (content-based routing) then you should reintroduce EBSs.

> **Note:** With the deployment of the Fusion Middleware Foundation Pack, web service definition language (WSDL) files are provided for all EBSs.

To reintroduce EBSs:

1. Go to JDevloper and create a new composite for the EBS with an Oracle Mediator service. Use the EBS WSDL provided by Fusion Middleware Foundation Pack.
2. Create routing rules in Oracle Mediator to route to appropriate Provider connectors.
3. Save your changes.
4. Open the AIAConfigurationProperties.xml file, which is located here: *$AIA_HOME/aia_instances/$INSTANCE_NAME/AIAMetaData/config*.

   Entries in the AIAConfigurationProperties.xml file are case sensitive.
5. For connectors that are to invoke the new EBS instead of directly invoking the Provider, replace the Provider connector's name and address with the name and address of the newly created EBS.

   This action tells the Requestor to invoke the EBS instead of directly invoking the Provider application business connector service (ABCS).
6. Save and close the file.
7. Login to the AIA Console (*http://<host>:<port>/AIA*). Go to **Setup**, and then select the **AIA Configuration** tab. Click **Reload** to relaod the configuration file and make your changes effective.