Siebel CRM Integration Pack for Oracle® Order Management: Order to Cash 2.5 - Implementation Guide

Release 2.5

Part No. E15769-01

October 2009

Siebel CRM Integration Pack for Oracle Order Management: Order to Cash 2.5 - Implementation Guide

Part No. E15769-01

Contents

Contents 1

Siebel CRM Integration Pack for Oracle Order Management: Order to Cash 2.5 - Implementation Guide Preface 5

Oracle Application Integration Architecture Foundation Pack Concepts and Technologies Guide 5

Oracle Application Integration Architecture - Foundation Pack: Integration Developer's Guide 6

Oracle Application Integration Architecture - Foundation Pack: Core Infrastructure Components Guide 7

Oracle Application Integration Architecture Process Integration Packs 7

Additional Resources 7

Chapter 1: Understanding the Siebel CRM Integration Pack for Oracle Order Management: Order to Cash 2.5 8

Siebel CRM Integration Pack for Oracle Order Management Process Integration Pack (Order to Cash: Siebel CRM – EBS PIP) 8

Order to Cash: Siebel CRM – EBS Business Process Flows 10

Order to Cash: Siebel CRM – EBS Solution Assumptions and Constraints 12

Chapter 2: Loading Initial Data 13

Understanding Initial Bulk Data Loads 13

Deploying ODI Repository Components 15

Loading Initial Customer Data 17

Loading Initial Product Data 23

Loading Initial Price List Data 25

Loading Initial Assets Data 32

Chapter 3: Understanding Process Integration for Customer Management 37

Process Integration for Customer Management 37

Synchronizing New Customer Accounts from Siebel CRM to Oracle EBS 40

Data Requirements 55

Siebel CRM Interfaces 56

Oracle EBS Interfaces 57

Core AIA Components 58

Integration Services 59

Chapter 4: Understanding Process Integration for Product Management 68

Process Integration for Product Management........ 68
Item Synchronization........ 72
Bill of Material Synchronization........ 76
Data Requirements........ 80
Siebel CRM Interfaces........ 80
Oracle EBS Interfaces........ 81
Core AIA Components........ 82
Integration Services........ 83
Chapter 5: Understanding Process Integration for Price Lists........ 89
Process Integration for Price Lists........ 89
Loading Initial Price Lists........ 90
Loading Incremental Price Lists........ 91
Chapter 6: Understanding Process Integration for Quotes........ 95
Process Integration for Quotes........ 95
Quotes Integration Flow........ 96
Data Requirements........ 97
Siebel CRM Interfaces........ 97
Oracle EBS Interfaces........ 98
Core AIA Components........ 98
Integration Services........ 99
Chapter 7: Understanding the Available to Promise Check Integration Flow........ 102
ATP Check Requests........ 102
Data Requirements........ 107
Siebel CRM Interfaces........ 107
Oracle EBS Interfaces........ 108
Core AIA Components........ 108
Integration Services........ 109
Chapter 8: Understanding the Shipping Charges Integration Flow........ 111
Shipping Charges Integration Flow........ 111
Data Requirements........ 115
Siebel CRM Interfaces........ 116
OTM Interfaces 116
Core AIA Components........ 116
Integration Services........ 117

Chapter 9: Understanding the Credit Check Integration Flow 119
Credit Check Requests 119
Data Requirements 123
Siebel CRM Interfaces 124
Oracle EBS Interfaces 124
Core AIA Components 124
Integration Services 125
Chapter 10: Understanding the Payment Authorization Integration Flow 127
Payment Authorization Requests 127
Data Requirements 132
Siebel CRM Interfaces 132
Oracle EBS Interfaces 133
Core AIA Components 133
Integration Services 134
Chapter 11: Understanding Process Integration for Order Management 136
Process Integration for Order Management 136
Sales Order Creation 140
Sales Order Updates (Oracle EBS Initiated) 145
Sales Order Revision (Siebel CRM Initiated) 151
Sales Order Cancellation 154
Siebel CRM Interfaces 156
Oracle EBS Interfaces 157
Core AIA Components 158
Integration Services 159
Chapter 12: Understanding Process Integration for Asset Management 173
Process Integration for Asset Management 173
Creating Assets 175
Updating Assets 179
Data Requirements 182
Siebel CRM Interfaces 182
Oracle EBS Interfaces 183
Core AIA Components 183
Integration Services 183
Chapter 13: Implementing the Order to Cash: Siebel CRM – EBS Process Integration Pack 186

Configuring Siebel CRM 186
Getting Inventory Location Details in Oracle EBS 189
Setting Up Organizations in Siebel CRM 190
Setting Up Cross-References for Siebel IDs and Oracle EBS Entities 192
Setting Up Application Context Definitions for Oracle EBS 195
Setting Up Oracle Configurator 196
Working with Cross-References 203
Working with DVMs 206
Enabling Oracle EBS Events 209
Creating Oracle EBS System Profiles 209
Scheduling Concurrent Processes 211
Setting a Property in OTM 211
Configuring the Payment Authorization Integration 211
Handling Errors 212
Viewing EBO Implementation Maps (EIMs) 213
Setting Configuration Properties 214
Performing Post-Installation Configurations 264
Appendix A 267
Configuring ODI-Based Initial Loads Against a Non-Oracle Target Database 267
Index 273

Siebel CRM Integration Pack for Oracle Order Management: Order to Cash 2.5 - Implementation Guide Preface

This preface discusses:

- The Oracle Application Integration Architecture - Foundation Pack: Concepts and Technologies Guide
- Oracle Application Integration Architecture - Foundation Pack: Integration Developer's Guide
- Oracle Application Integration Architecture - Foundation Pack: Core Infrastructure Components Guide
- Oracle Application Integration Architecture Process Integration Packs
- Additional resources

Oracle Application Integration Architecture Foundation Pack Concepts and Technologies Guide

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

- Oracle AIA.
- Enterprise business objects and enterprise business messages.
- Enterprise business services.
- Application business connector services.
- Interaction patterns.
- Extensibility.
- Versioning.
- Business processes.
- Batch processing.
- Infrastructure services.

- Security.

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

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

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

- Create an integration scenario.
- Define business service patterns.
- Design and develop enterprise business services.
- Design and develop enterprise business flows.
- Design and construct Application Business Connector Services (ABCS).
- Work with message transformation, enrichment, and configuration.
- Develop custom XPath functions.
- Design and construct JMS Adapter services.
- Work with enterprise message headers.
- Work with message routing.
- Work with transactions.
- Develop Oracle Application Integration Architecture (AIA) services to work with the CAVS.
- Configure Oracle AIA processes to be eligible for error handling and logging.
- Extend enterprise business objects (EBOs).

In addition, this book provides:

- AIA naming standards.
- Sample and template WSDLs for use with Oracle AIA.

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

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

Oracle Application Integration Architecture - Foundation Pack: Core Infrastructure Components Guide discusses how to:

- Work with the Composite Application Validation System (CAVS).
- Work with the Business Service Repository (BSR).
- Set up and use error handling and logging.
- Work with the diagnostics framework.

Oracle Application Integration Architecture Process Integration Packs

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

Additional Resources

The following resources are available:

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

Chapter 1: Understanding the Siebel CRM Integration Pack for Oracle Order Management: Order to Cash 2.5

This chapter discusses:

- Siebel CRM Integration Pack for Oracle Order Management Process Integration Pack (Order to Cash: Siebel CRM – EBS PIP)
- Order to Cash: Siebel CRM – EBS business process flows
- Order to Cash: Siebel CRM – EBS solution assumptions and constraints

Siebel CRM Integration Pack for Oracle Order Management Process Integration Pack (Order to Cash: Siebel CRM – EBS PIP)

The Siebel CRM Integration Pack for Oracle Order Management Process Integration Pack (PIP) provides a seamless and robust order-to-cash business process. Leveraging the best front office with the best back office applications, this PIP provides a solution that gives a streamlined, end-to-end Order to Cash: Siebel CRM – EBS business process that enables faster time to market for new products and 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 bi-directionally, while product (item or bill of material [BOM]) data is synchronized from the Oracle EBS product master.

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

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

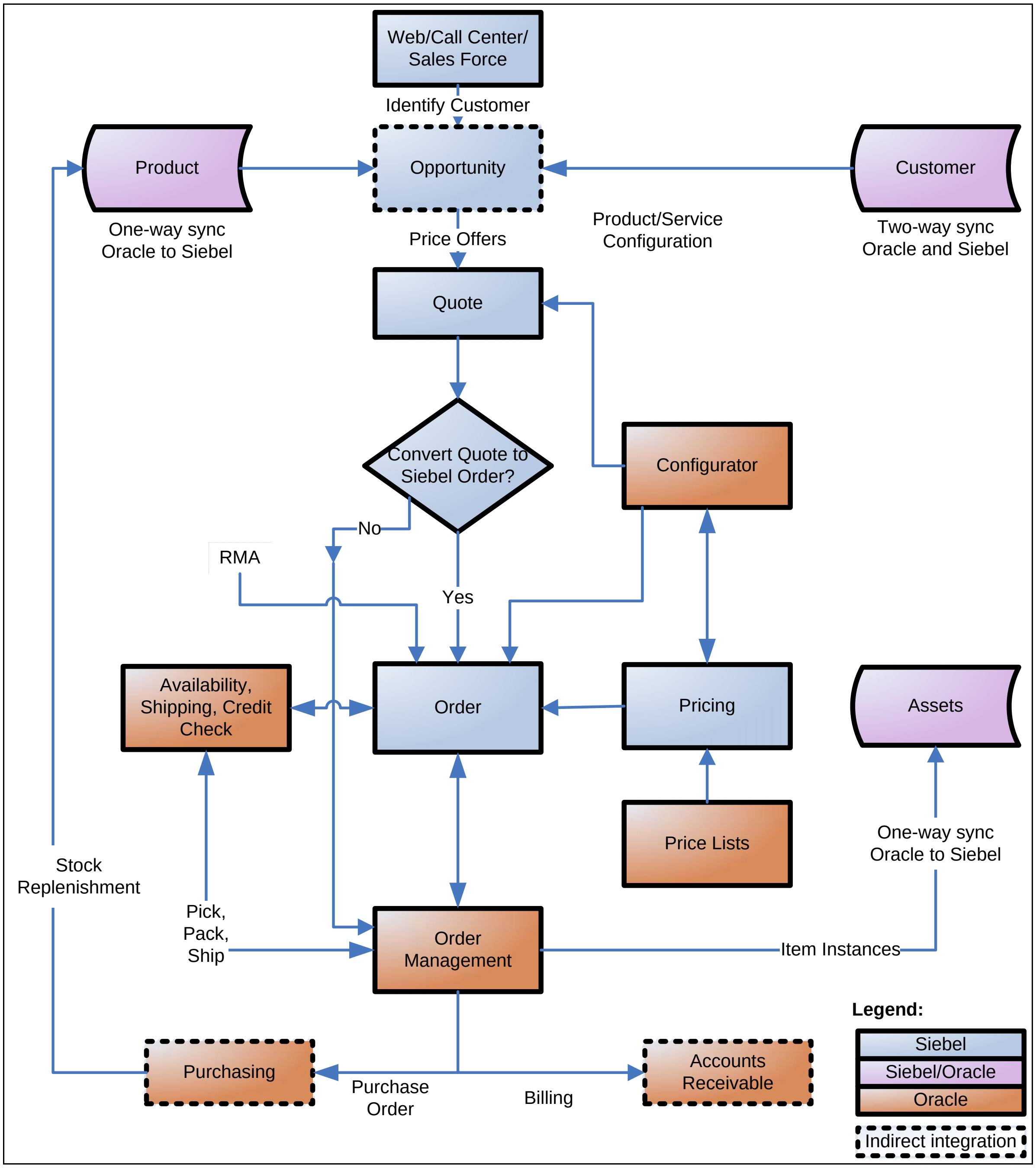
The integration pack also provides the ability to invoke the Oracle Configurator, either from a Siebel quote or an order. This approach eliminates the costly and complex maintenance of configuration rules in CRM and ERP. It also leverages a customer's existing investment in Oracle Configurator.

This business process helps an organization:

- Eliminate order fallout by making sure that products are correctly configured and available before the order is confirmed.
- Increase customer satisfaction by making sure 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 by means of delivered mappings and business processes.

This diagram illustrates the Order to Cash: Siebel CRM – EBS PIP solution:



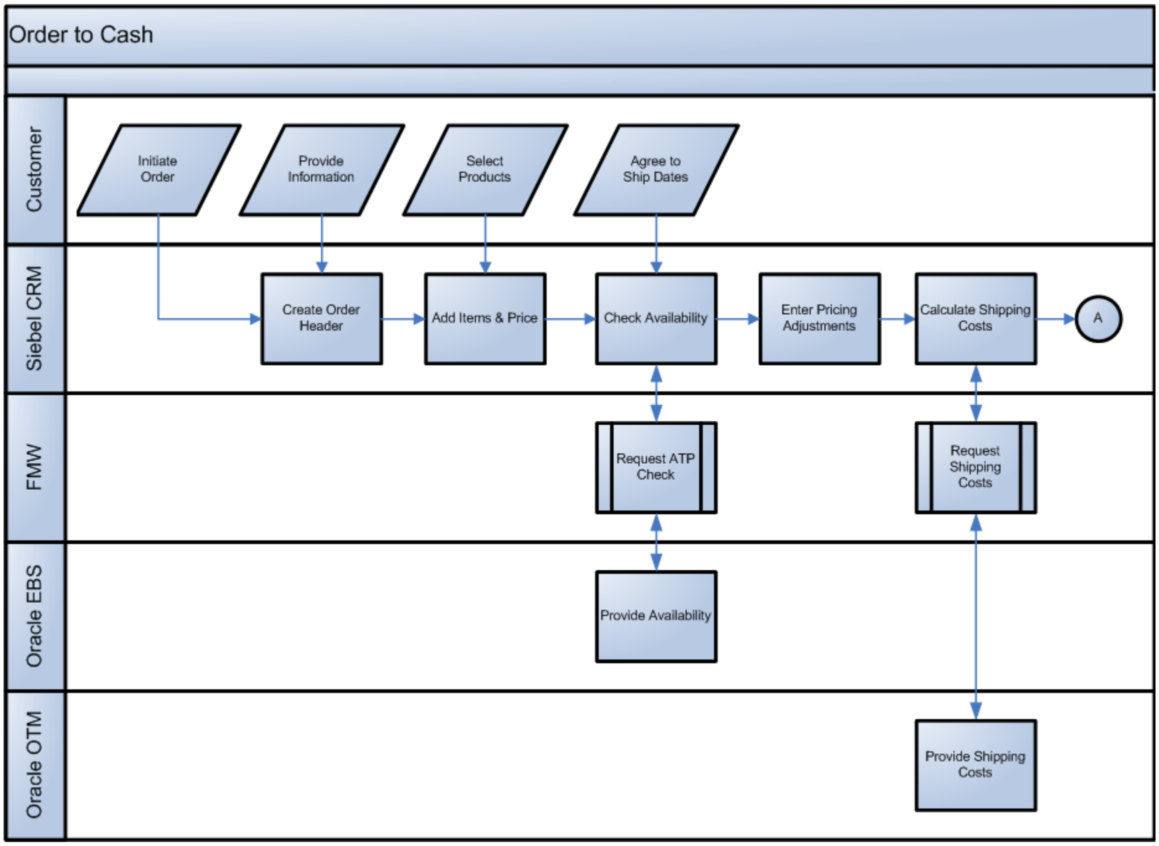
Order to Cash: Siebel CRM – EBS Process Integration Pack

Order to Cash: Siebel CRM – EBS Business Process Flows

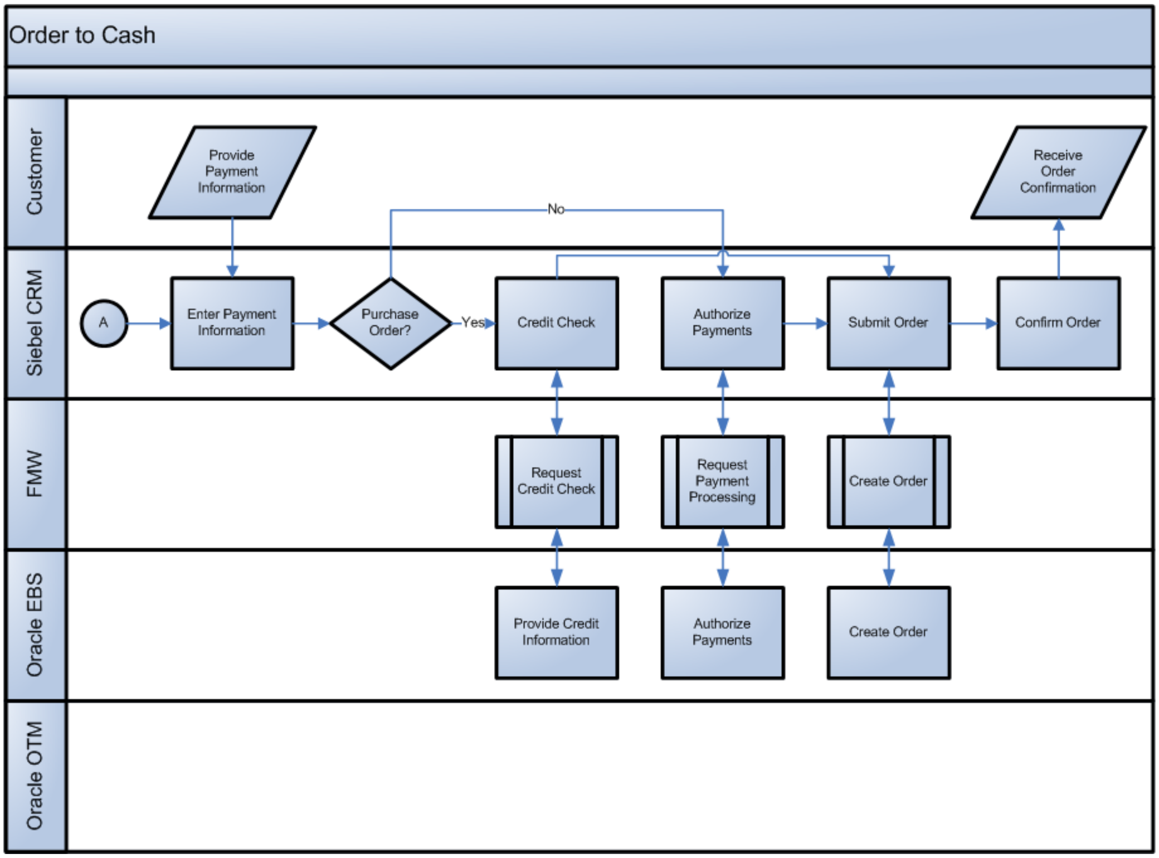
The Order to Cash: Siebel CRM – EBS PIP consists of these integration flows:

- 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

These diagrams illustrate the Order to Cash: Siebel CRM – EBS business process flow:



Order to Cash: Siebel CRM – EBS business process flow (1 of 2)



Order to Cash: Siebel CRM – EBS business process flow (2 of 2)

The Order to Cash: Siebel CRM – EBS business process allows companies 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 calculations, and multiple methods of payment to ensure that orders are complete and accurate before they are delivered to the customer. The initial data load processes enable bulk loading of customers, products, price lists, and assets.

In the Order to Cash: Siebel CRM – EBS process, customers contact customer service representatives (CSRs) to place new orders. CSRs must first determine if the customer is existing or new. For new customers, CSRs must capture all of the pertinent information related to them and create accounts in Siebel CRM. Next, the CSR creates orders. The order header is created first, and then the order line items are created. By working with the customer, products are selected and shipping information is entered.

You should perform an 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, you should also perform a credit check to ensure that an order will not be placed on hold for credit reasons. Optionally, it is 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. Once the order is submitted, the CSR cannot make further modifications to it in Siebel CRM.

Order updates that happen in the back office as a result of order fulfillment are sent to Siebel CRM. The CSR is able to access the latest order status, thus serving customers better without having to toggle to a different application.

Order to Cash: Siebel CRM – EBS Solution Assumptions and Constraints

The assumptions and constraints for the Order to Cash: Siebel CRM – EBS PIP are:

1. Delivered support is for Siebel SIA version only.
2. Delivered configurator support is for Oracle Configurator.
3. Oracle EBS is the product master, and the item/BOM synchronization happens only one-way: from Oracle EBS to Siebel CRM.
4. Pricing is done in Siebel CRM, and Oracle EBS does not recalculate the price.
5. B2C order scenarios are not supported without customization.
6. Prospects are not synchronized to Oracle EBS.
7. Change orders and cancel orders are not supported without customization.
8. Participating applications are running in either the same code page or Unicode.

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

Chapter 2: Loading Initial Data

This chapter provides an overview of initial bulk data loads and discusses how to:

- Deploy Oracle Data Integration (ODI) repository components
- Load initial customer data
- Load initial product data
- Load initial price list data
- Load initial assets data

Understanding Initial Bulk Data Loads

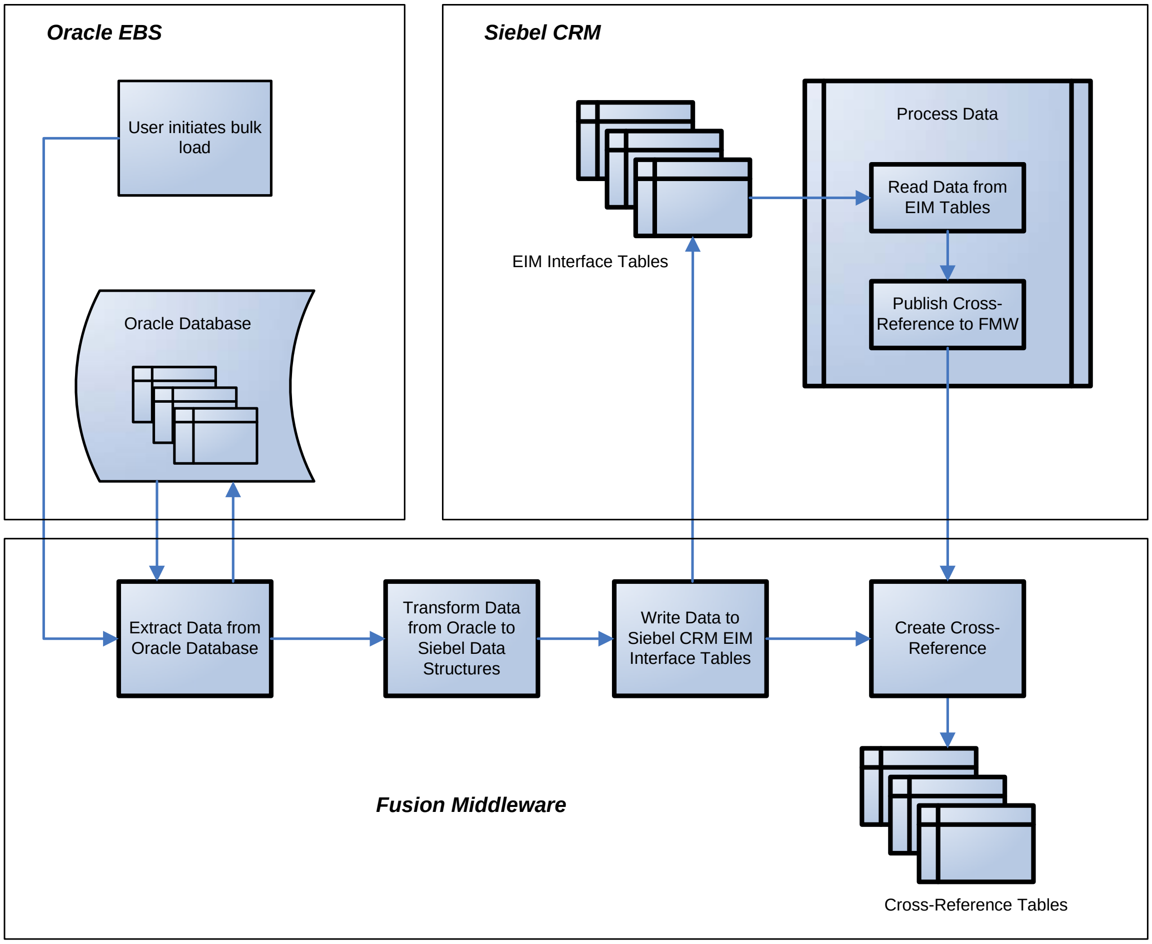
Siebel CRM Integration Pack for Oracle Order Management requires that the business data in the Oracle and Siebel databases is always synchronized. Siebel CRM Integration Pack for Oracle Order Management lets you:

- 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. Using Oracle Data Integrator (ODI), business data is extracted from Oracle E-Business Suite (Oracle EBS), transformed into Siebel database-compatible structure and format, and imported into 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.

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

This diagram illustrates the bulk load process:


Oracle EBS
User initiates bulk load
Oracle Database
Siebel CRM
EIM Interface Tables
Process Data
Read Data from EIM Tables
Publish Cross-Reference to FMW
Extract Data from Oracle Database
Transform Data from Oracle to Siebel Data Structures
Write Data to Siebel CRM EIM Interface Tables
Create Cross-Reference
Fusion Middleware
Cross-Reference Tables

Loading bulk data from Oracle EBS to Siebel CRM

Prerequisites

Before proceeding with the data loads, ensure that:

- The Siebel CRM Integration Pack for Oracle Order Management Process Integration Pack (PIP) has been installed.
- ODI is installed on the same machine as AIA_HOME.
- ODI master and work repositories are created.
- Siebel is installed on Oracle, and users have access to the Siebel base tables and EIM tables for account, product, asset, and price list.
- 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.

Solution Assumptions and Constraints

1. Price lists are dependent on product data.

   Assets are dependent on customer and product data. Make sure 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 causes Asset initial load to fail due to ODI is unable to create a temporary table because this value exceedes the allowed limit for the target field in Siebel CRM. Thus, before importing the data to Siebel CRM, this significantly high number should be set to a number that can be put on the Siebel quantity attribute. In Siebel CRM, the maximum value the Asset quantity field may hold is E+11.

For more information about organizations and inventory locations, see Chapter 13: Implementing the Order to Cash: Siebel CRM – EBS Process Integration Pack

Deploying ODI Repository Components

Complete the following steps to deploy the Order to Cash: Siebel CRM – EBS 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 make sure the data server connection works.

In UNIX

- Set ODI_HOME.
  Example: export ODI_HOME=$HOME/odi/oracledi
- Set the environment variable to:
  Source: aiaenv.sh from AIA_HOME/bin
- Set up the Repository connection information in ODI_HOME/bin/ odiparams.sh as follows:

```
ODI_SECU_DRIVER=oracle.jdbc.driver.OracleDriver
ODI_SECU_URL=jdbc:oracle:thin:@<hostname>:<port>:<sid>
ODI_SECU_USER=<masterrepuser>
ODI_SECU_ENCODED_PASS=<encodedmasterreppassword>
ODI_SECU_WORK_REP=<workrepuser>
ODI_USER=SUPERVISOR
ODI_ENCODED_PASS=fJya,N15,jlR.EM64bpAAOo
```

- Use this command to encrypt a password from ODI_HOME/bin: sh agent.sh encode odimaster

Instructions for Deployment

To deploy components for all bulk loads:

1. Navigate to AIA_HOME/PIPS/Core/Setup/Install/O2C2.
2. Run: ant –buildfile O2C2ODIRepinstall.xml

To deploy components for individual bulk loads:

1. Navigate to AIA_HOME/PIPS/Core/Setup/Install/O2C2.
2. For Customer, run: ant –buildfile ODICustomerInstall.xml
3. For Product, run: ant –buildfile ODIProductInstall.xml
4. For Price List, run: ant –buildfile ODIPriceListInstall.xml
5. For Asset, run: ant –buildfile ODIAssetsInstall.xml

In Windows

- Set ODI_HOME.

  Example: set ODI_HOME=D:\odi\oracledi

- Set env as:

  aiaenv.bat from AIA_HOME\bin

- Set up the repository connection information in ODI_HOME\bin\ odiparams.sh as follows:

```
ODI_SECU_DRIVER=oracle.jdbc.driver.OracleDriver
ODI_SECU_URL=jdbc:oracle:thin:@<hostname>:<port>:<sid>
ODI_SECU_USER=<masterrepuser>
ODI_SECU_ENCODED_PASS=<encodedmasterreppassword>
ODI_SECU_WORK_REP=<workrepuser>
ODI_USER=SUPERVISOR
ODI_ENCODED_PASS=fJya,N15,jlR.EM64bpAAOo
```

- Use this command to encrypt a password from ODI_HOME\bin: agent encode odimaster

Instructions for Deployment

To deploy components for all bulk loads:

1. Navigate to AIA_HOME\PIPS\Core\Setup\Install\O2C2
2. Run: ant –buildfile O2C2ODIRepinstall.xml

To deploy components for individual bulk loads:

1. Navigate to AIA_HOME\PIPS\Core\Setup\Install\O2C2
2. For Customer, run: ant –buildfile ODICustomerInstall.xml
3. For Product, run: ant –buildfile ODIProductInstall.xml
4. For Price List, run: ant –buildfile ODIPriceListInstall.xml
5. For Asset, run: ant –buildfile ODIAssetsInstall.xml

Setting Up 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

Loading Initial Customer Data

Siebel CRM Integration Pack for Oracle Order Management provides you with a bulk load feature to move data from Oracle E-Business Suite into Siebel CRM.

The bulk load feature extracts customer data from Oracle EBS, transforms it into EIM interface table data structures, and moves it into the Siebel EIM interface tables. Siebel EIM tables are intermediate interface tables that act as a staging area between the base tables in 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 with a relationship to 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.

Loading Customer and Contact Bulk Data

The Customer bulk load consists of these steps:

1. Load Oracle EBS to Seibel 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 table. The 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 AIA cross-reference table.

   Use ODI to move the Siebel ID values back into the AIA cross-reference table, correlating to the Oracle EBS and Common ID values that were populated in Step 1.

Some EIM activities must be performed for initial data loads. **For more information** about the pre- and post-EIM activities and execution, see the Siebel CRM Integration Pack for Oracle Order Management Addendum

Moving Data from Oracle EBS to Siebel EIM Tables

Perform these actions from 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.
7. Enter the following command to carry out the step:

   For EBusiness Suite 12.1.1: Go to the bin folder of ODI in the command prompt

   ```
   Windows: startscen LOAD_EBIZ_ASSET_DATA_PKG_R12 001 GLOBAL
   ```

   ```
   UNIX: ./startscen.sh LOAD_EBIZ_ASSET_DATA_PKG_R12 001 GLOBAL
   ```

   For EBusiness Suite 11.5.10: Go to the bin folder of ODI in the command prompt

   ```
   Windows: startscen LOAD_EBIZ_ASSET_DATA_TO_EIM_PKG 001 GLOBAL
   ```

   ```
   UNIX: ./startscen.sh LOAD_EBIZ_ASSET_DATA_TO_EIM_PKG 001 GLOBAL
   ```

Moving 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.

   Example: in the D:\ses\siebsrvr\Admin directory

2. Log in to the Siebel E-Business Applications using the Administrator login.

   Example: proper credentials like 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:

   a Configuration File: Customer.ifb

   b Username: The administrator username ex. SADMIN

   c Password: The administrator password ex. SADMIN

   d Error Flag: 1

   e SQL Trace Flag: 8

   f Trace Flag: 1

9. Ensure that the job is completed with a status of Success.

   Color of status bar changes to green.

10. 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.
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 following status values:

    IF_ROW_STAT = for import: Job to be executed or Job in execution.

    IF_ROW_STAT = Partially_imported: Some mandatory field missing for that record.

    IF_ROW_STAT = Imported: Job executed successfully for that record.

    IF_ROW_STAT= dup_row_exists: Siebel base table already has that record.

Moving Data from Siebel Base Tables to AIA Cross-Reference Tables

Perform these actions from 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 Siebel Data to XREF Pkg and select Execute from the menu.
6. Select context as the Global in the execution box.

   Enter the following command to run the step.

   Go to the bin folder of ODI in the command prompt:
   Windows: Startscen LOAD_SIEBEL_DATA_TO_XREF_PKG 001 GLOBAL
   UNIX:./startscen.sh LOAD_SIEBEL_DATA_TO_XREF_PKG 001 GLOBAL

To verify the data:

After the process completes and before moving data from the Siebel EIM table to the Siebel base table, verify that the Asset 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='INSTALLEDPRODUCT_ID' GROUP BY xref_column_name (to be run in FMW DB).
2. Select count(*) from eim_asset (to be run in Siebel DB).

   The count should be the same.

3. Verify that the AIA cross-reference tables are populated with the Oracle EBS and Common ID values for the Asset.
4. After Moving data from the Siebel EIM table to the Siebel base table verify that the Asset data was loaded into the Siebel base table by completing these actions:

   Select count(*) from s_asset (to be run in Siebel DB).

5. After Moving data from the Siebel base table to the AIA cross-reference table, verify that the Siebel ID is populated by completing this action:

   > Select xref_column_name,count(*) from xref_data where xref_table_name='INSTALLEDPRODUCT_ID' GROUP BY xref_column_name (to be run in FMW DB).

6. Verify that the AIA cross-reference tables are populated with the Oracle EBS, Common, and Siebel ID values for Asset.

Verifying Data After the Load

Use this table to verify the data:

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,	Number of Active Accounts in Oracle EBS

Entity	Database	Query	Inference
		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);	
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.
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_ROLE	Number of Contacts of the Active Accounts and Parties.

Entity	Database	Query	Inference
		S.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)	
CONTACTS	SIEBEL	select count(*) from s_contact where db_last_upd_src='EIM';	Number of Contacts successfully transferred to Siebel from the EIM job.
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.

Loading Initial Product Data

Use the bulk load feature to move product data from Oracle E-Business Suite into Siebel CRM.

Bulk loading product data is similar to bulk loading customer data. The bulk load feature extracts product data from Oracle E-Business Suite, 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.

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 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 AIA cross-reference table.

   Use ODI to move the Siebel ID values back into the AIA cross-reference table, correlating to the Oracle EBS and Common ID values that were populated in Step 1.

Some EIM activities must be performed for initial data loads. **For more information** about the pre- and post-EIM activities and execution, see the Siebel CRM Integration Pack for Oracle Order Management Addendum

Moving Data from Oracle EBS to Siebel EIM Tables

Perform these actions from 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 Load_Oracle_Ebiz_Product_To_Siebel and select Execute from the menu.
6. Select the Product Development context in the execution box.
7. Enter the following command to run the step:

   Go to the bin folder of ODI in the command prompt:

   Windows: Startscen LOAD_ORACLE_EBIZ_PRODUCT_TO_SIEBEL 001 PRODUCTDEVELOPMENT

UNIX: ./startscen.sh LOAD_ORACLE_EBIZ_PRODUCT_TO_SIEBEL 001 PRODUCTDEVELOPMENT

Moving 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. Log in to the Siebel E-Business Applications using the Administrator login.

   For example, use proper credentials like 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:

   a Configuration File: Product.ifb

   b Username: The administrator username ex. SADMIN

   c Password: The administrator password ex. SADMIN

   d Error Flag: 1

   e SQL Trace Flag: 8

   f Trace Flag: 1

9. Ensure that the job is completed with a status of Success.

   Color of the status bar changes to green.

10. Release the products after getting the product data imported into the base table.

    See the EIM documentation provided by Siebel.

11. Ensure that the data appears correctly 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 the following status values:

    a IF_ROW_STAT = for import: Job to be executed OR Job in execution.

    b IF_ROW_STAT = Partially_imported: Some mandatory field missing for that record.

    c IF_ROW_STAT = Imported: Job executed successfully for that record.

    d IF_ROW_STAT= dup_row_exists: Siebel base table already has that record.

Moving Data from Siebel Base Tables to AIA Cross-Reference Tables

Perform these actions from 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 Load_Siebel_To_XREF_Data and select Execute from the menu.
6. Select context as the Product Development in the execution box.
7. Enter the following command to carry out the step.

   Go to the bin folder of ODI in the command prompt:

   Windows: Startscen LOAD_SIEBEL_TO_XREF_DATA 001 PRODUCTDEVELOPMENT

   UNIX: ./startscen.sh LOAD_SIEBEL_TO_XREF_DATA 001 PRODUCTDEVELOPMENT

To verify the data:

After the process completes and before moving data from the Siebel EIM table to the Siebel base table, verify 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 = 'ITEM_ITEMID' 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 same.

5. Verify that the AIA cross-reference tables are populated with the Oracle EBS and Common ID values for 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 moving data from the Siebel EIM table to the Siebel base table.
7. Select xref_column_name, count(*) from xref_data where xref_table_name = 'ITEM_ITEMID' GROUP BY xref_column_name (to be run in FMW DB) to verify that the Siebel ID is populated after moving data from the Siebel base table to the AIA cross-reference table.
8. Verify that the AIA cross-reference tables are populated with the Oracle EBS, Common, and the Siebel ID values for product.

Loading Initial Price List Data

Use the bulk load feature to move active price list 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, could 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 Siebel CRM database, depending on whether or not the price lists or the header and line information already exists in the Siebel CRM database.

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 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 AIA cross-reference table.

   Use ODI to move the Siebel ID values back into the AIA cross-reference table (XREF_DATA), correlating to the Oracle EBS and Common ID values that were populated in Step 1.

Some EIM activities must be performed for initial data loads. **For more information** about the pre- and post-EIM activities and execution, see the Siebel CRM Integration Pack for Oracle Order Management Addendum

Moving Data from Oracle EBS to Siebel EIM Tables

To move data from EBS to Siebel EIM, run the ODI package either from the command line or from the ODI designer.

Perform these actions from the command line:

1. To run from command line, run the following ODI scenario from ODI_HOME/bin:

   LINUX: ./startscen.sh LOADEBIZPRICELISTDATATOSIEBEL 001 PRICELISTBULKLOAD

   Windows: startscen.bat LOADEBIZPRICELISTDATATOSIEBEL 001 PRICELISTBULKLOAD

Perform these actions from the ODI Designer:

1. Log in to the ODI Designer.
2. Under the project PriceListBulkLoad, run the package LoadEbizPriceListDataToSiebelPkg.

To verify the data:

1. After the process completes, 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).

   The count should be the same.

2. Verify that the 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 like ‘PRICELIST_ID’ and xref_column_name like ‘EBIZ_01’.

   c Select count(*) from xref_data where xref_table_name like ‘PRICELIST_ID’ and xref_column_name like ‘COMMON’.

   The count should be the same.

Moving 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. For Pricelist.ifb, comment the second process using a terminate symbol (so that it becomes inactive and only the first process is run) such as:

   [IMPORT ALL PRICELISTS]

   TYPE = SHELL

   INCLUDE = "IMPORT PRICELIST"

   INCLUDE = "UPDATE INTEGRATION ID"

3. Log in to the 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:

   a Configuration File: Pricelist.ifb

   b Username: The administrator username ex. SADMIN

   c Password: The administrator password ex. SADMIN

   d Error Flag: 1

   e SQL Trace Flag: 8

   f Trace Flag: 1

10. Ensure that the job is completed with a status of Success.

    Color of the status bar changes to green.

11. Ensure that the Siebel Business Applications User Interface displays the data properly 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:

    a IF_ROW_STAT = for import: Job to be executed OR Job in execution.

    b IF_ROW_STAT = Partially_imported: Some mandatory field missing for that record.

    c IF_ROW_STAT = Imported: Job executed successfully for that record.

    d IF_ROW_STAT= dup_row_exists: Siebel base table already has that record.

Moving Data from Siebel Base Tables to AIA Cross-Reference Tables

To create the cross-reference for the Siebel Price List IDs, the ODI package can be run from the command line or from the ODI designer.

To run from the command line:

1. From the ODI_HOME/bin directory:

   LINUX: ./startscen.sh LOADSIEBELPRICELISTDATATOXREF 001 PRICELISTBULKLOAD

   Windows: startscen.bat LOADSIEBELPRICELISTDATATOXREF 001 PRICELISTBULKLOAD

To run from the ODI designer:

1. Log in to the ODI Designer.

2. Under the project PriceListBulkLoad, run the package LoadSiebelPriceListDataToXREFPkg.

To verify the data after load:

After the process completes, the cross references for the Siebel Price List IDs are created. Perform these actions to verify:

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 like ‘PRICELIST_ID’ and xref_column_name like ‘SEBL_01’

   The count should be the same.

Note: After the ODI package to populate the X-REF Database is complete, if needed, 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 will look like:

```
[IMPORT ALL PRICELISTS]
TYPE = SHELL
;INCLUDE = "IMPORT PRICELIST"
INCLUDE = "UPDATE INTEGRATION ID"
```

All the other job parameters will remain the same in the ifb file.

Loading Incremental Price Lists

The three steps to the incremental load include:

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/staging/O2C2Home/SetupFiles/PriceListSource.xml.

   The 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. Populate the Siebel ID values in the AIA cross-reference table (XREF_DATA).

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, run the following ODI scenario from the directory ODI_HOME/bin either from the command line or from the ODI Designer:

To run from the command line:

1. LINUX: ./startscen.sh LOADEBIZINCPRICELISTDATATOSIEBEL 001 PRICELISTBULKLOAD

   Windows: startscen.bat LOADEBIZINCPRICELISTDATATOSIEBEL 001 PRICELISTBULKLOAD

To run from the ODI Designer:

1. Log in to the ODI Designer.
2. Under the project PriceListBulkLoad, run the package LoadEbizIncPriceListDataToSiebelPkg.

To verify the data after load:

After the process completes, 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, select a unique PL_NAME from eim_pri_lst (Siebel DB).
   The price list names should match with the ones in the XML file.
2. To verify that the 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 make sure the IDs returned are in the cross-reference table.

Running 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) such as:

   ```
   [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 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:

   a Configuration File: The exact name of the ifb file, for example, Product.ifb, Pricelist.ifb and so on.

   b Username: The administrator username, for example, SADMIN

   c Password: The administrator password, for example, SADMIN d Error Flag: 1

e SQL Trace Flag: 8

f Trace Flag: 1

10. Ensure that the job is completed 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:

    a IF_ROW_STAT = for import: Job to be executed OR Job in execution.

    b IF_ROW_STAT = Partially_imported: A mandatory field is missing for that record.

    c IF_ROW_STAT = Imported: Job executed successfully for that record.

    d IF_ROW_STAT= dup_row_exists: Siebel base table already has that record.

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, run the ODI scenario from ODI_HOME/bin either from the command line or from the ODI Designer.

To run from the command line:

1. LINUX: ./startscen.sh LOADSIEBELINCPRICELISTDATATOXREF 001 PRICELISTBULKLOAD

   Windows:startscen.bat LOADSIEBELINCPRICELISTDATATOXREF 001 PRICELISTBULKLOAD

To run from the ODI Designer:

1. Log in to ODI Designer.
2. Under the project PriceListBulkLoad, run the package LoadSiebelIncPriceListDataToXREFPkg.

To verifying data after load:

1. After the process completes, verify that the cross-references for the Siebel Price List IDs created by selecting row_id from s_pri_lst where the pricelist names in the xml file ($AIA_HOME/staging/O2C2Home/SetupFiles/PriceListSource.xml) (to be run in Siebel database).
2. Make sure the IDs returned are created in the cross-reference table.

Note: After the ODI package to populate the X-REF Database is complete, 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 will look like this:

```
[IMPORT ALL PRICELISTS]
TYPE = SHELL
;INCLUDE = "IMPORT PRICELIST"
INCLUDE = "UPDATE INTEGRATION ID"
```

All the other job parameters will remain the same in the ifb file.

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.

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 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 AIA cross-reference table.

   Use ODI to move the Siebel ID values back into the AIA cross-reference table (XREF_DATA), correlating to the Oracle EBS and Common ID values that were populated in Step 1.

Some EIM activities must be performed for initial data loads. **For more information** about the pre- and post-EIM activities and execution, see the Siebel CRM Integration Pack for Oracle Order Management Addendum

Moving Data from Oracle EBS to Siebel EIM Tables

Perform these actions from the 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.
7. Enter the following command to run the step:

   Go to the bin folder of ODI in the command prompt

   Windows: Startscen LOAD_EBIZ_ASSET_DATA_TO_EIM_PKG 001 GLOBAL

   UNIX: ./startscen.sh LOAD_EBIZ_ASSET_DATA_TO_EIM_PKG 001 GLOBAL

Moving 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, into the D:\ses\siebsrvr\Admin directory

2. Log in to the Siebel E-Business Applications using the Administrator login.

   For example, proper credentials like 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:

   a Configuration File: Assets.ifb

   b Username: The administrator username ex. SADMIN

   c Password: The administrator password ex. SADMIN

   d Error Flag: 1

   e SQL Trace Flag: 8

   f Trace Flag: 1

9. Ensure that the job is completed with a status of Success.

   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 like 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 following status values:

a IF_ROW_STAT = for import: Job to be executed OR Job in execution.

b IF_ROW_STAT = Partially_imported: A mandatory field is missing for that record.

c IF_ROW_STAT = Imported: Job executed successfully for that record.

d IF_ROW_STAT= dup_row_exists: Siebel base table already has that record.

Moving Data from Siebel Base Tables to AIA Cross-Reference Tables

To run from the 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 Siebel Data to XREF Pkg and select Execute from the menu.
6. Select context as the Global in the execution box. Enter the following command to run the step.
7. Go to the bin folder of ODI in the command prompt:

   Windows: Startscen LOAD_SIEBEL_DATA_TO_XREF_PKG 001 GLOBAL

   UNIX: ./startscen.sh LOAD_SIEBEL_DATA_TO_XREF_PKG 001 GLOBAL

To verify the data:

1. After the process completes 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 on execution of first ODI scenario of Asset Initial load:

   Select count(*),Xref_column_name from xref_data

   Where xref_table_name='INSTALLEDPRODUCT_ID' group by xref_column_name. Count of 'COMMON' and 'EBIZ_01' should be same.

2. Run this query in the siebel database (after the execution of 2nd ODI scenario) to count the number of records in the EIM table of the Siebel:

   Select count(*),Xref_column_name from xref_data

   Where xref_table_name='INSTALLEDPRODUCT_ID' group by xref_column_name. This will give the count of Ebiz records loaded into siebel, along with their Common IDs count. The count should match.

   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).

3. If the load failed before the data is loaded into the Siebel base tables (through EIM jobs), you can directly go 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.

4. If it fails after running the EIM job, you need to clean up the data from the Siebel database tables and then continue with the step 1.

The IF_ROW_STAT column can have these statuse: 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 means that the data is already present in the Siebel database so that the record is not loaded into the Siebel base tables. Similarly, failed means inport of that record is failed and partially failed means some inconsistency in the data in the EIM table.

Chapter 3: Understanding Process Integration for Customer Management

This chapter discusses:

- Process integration for Customer Management
- Synchronization of new customer accounts from Siebel Customer Relationship Management (CRM) to Oracle E-Business Suite (Oracle EBS)
- Data requirements
- Siebel Customer Relationship Management (CRM) interfaces
- Oracle E-Business Suite (Oracle EBS) interfaces
- Core Oracle Application Integration Architecture (AIA) components
- Integration services

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** enables the synchronization of new customer accounts from Siebel CRM to Oracle EBS as part of booking an order in the process integration for Order Management.
- **Update and synchronize customer accounts from Siebel CRM to Oracle EBS** enables the synchronization of customer updates from Siebel CRM to Oracle EBS if the record exists in both Siebel CRM and Oracle EBS.

  If a customer record that does not exist in Oracle EBS is updated, the customer record will not be synchronized to Oracle EBS.
- **Synchronize customer accounts from Oracle EBS to Siebel CRM** enables the synchronization of customer accounts from Oracle EBS to Siebel CRM.

  Whenever a new customer account is created in Oracle EBS, a realtime synchronization flow is initiated to synchronize the related party and customer account to a new Siebel customer account.
- **Merge accounts** maintains the transactional integrity essential to convert orders captured in Siebel CRM into booked orders, despite customer merge events that may occur within Oracle EBS.

  This flow is initiated in Oracle EBS when a customer account is merged into another customer account. The merge operation is then performed in Siebel CRM.

- **Merge parties** 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** 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, and load it to the appropriate Siebel CRM EIM tables, to be inserted 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: Describing the Initial Data Loads

Prerequisites

The Customer Management process integration does not have a dependency 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 13: Implementing the Order to Cash: Siebel CRM – EBS Process Integration Pack

Solution Assumptions and Constraints

The integration design assumes that the following statements are true:

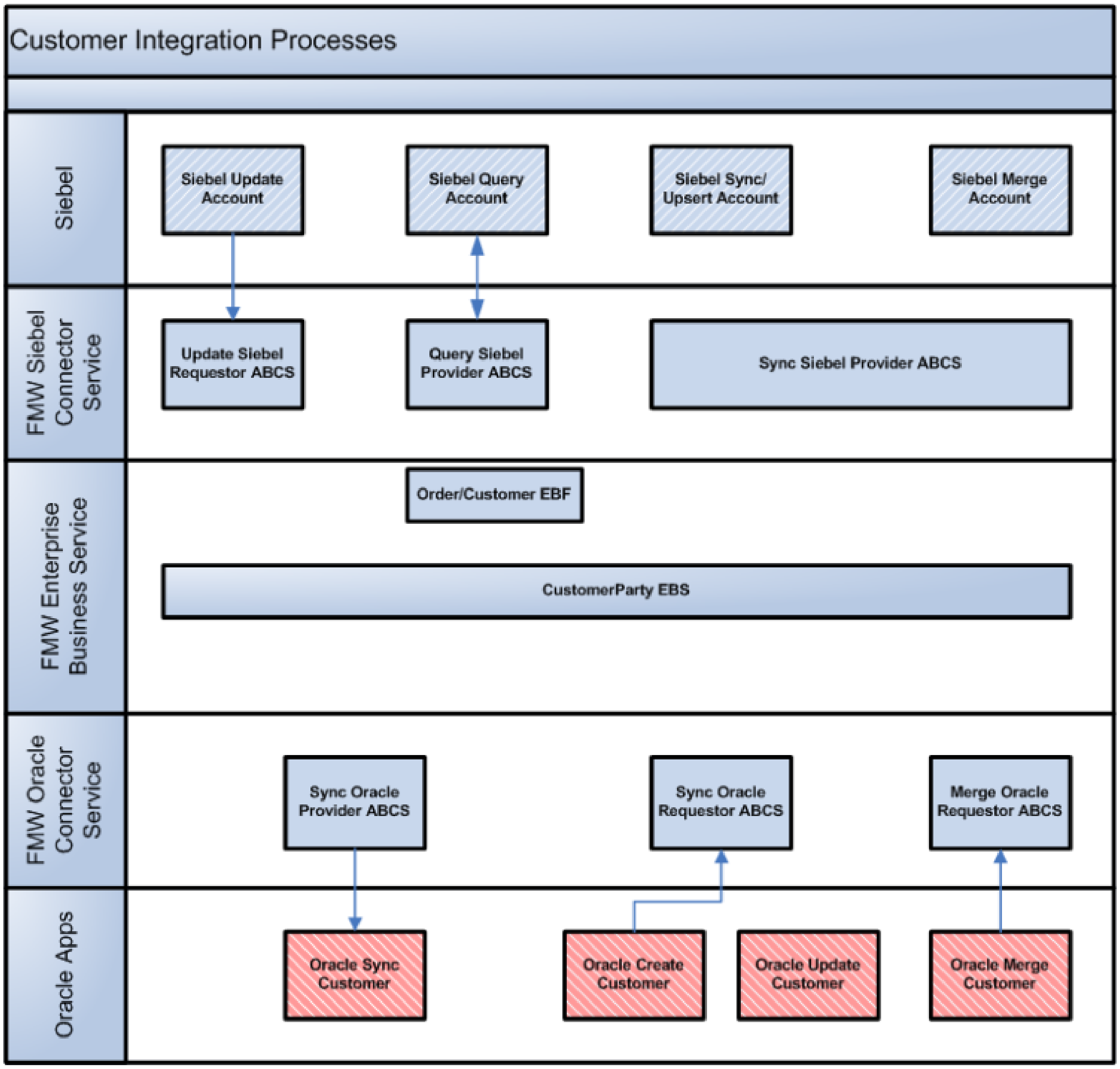
1. Account hierarchy is not in the scope of this release.
2. To initiate the Customer synchronization, users must run a concurrent job after any creates or updates to accounts or parties.

   A concurrent job is a bulk job that raises events for creates or updates since the last concurrent job.

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 auto-generated in Oracle EBS.

This diagram illustrates the overall flow for the customer process integration:


Customer Integration Processes
Siebel
FMW Siebel Connector Service
FMW Enterprise Business Service
FMW Oracle Connector Service
Oracle Apps
Siebel Update Account
Siebel Query Account
Siebel Sync/ Upsert Account
Siebel Merge Account
Update Siebel Requestor ABCS
Query Siebel Provider ABCS
Sync Siebel Provider ABCS
Order/Customer EBF
CustomerParty EBS
Sync Oracle Provider ABCS
Sync Oracle Requestor ABCS
Merge Oracle Requestor ABCS
Oracle Sync Customer
Oracle Create Customer
Oracle Update Customer
Oracle Merge Customer

Customer process integration flow

Customer information related to orders is maintained in both Siebel CRM and Oracle EBS, and is synchronized bi-directionally. New accounts can be created in Siebel CRM and maintained. An account is synchronized with 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 TCA Manager responsibility. The synchronize flow is initiated after this request completes successfully. The corresponding primary and bill to/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/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 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.

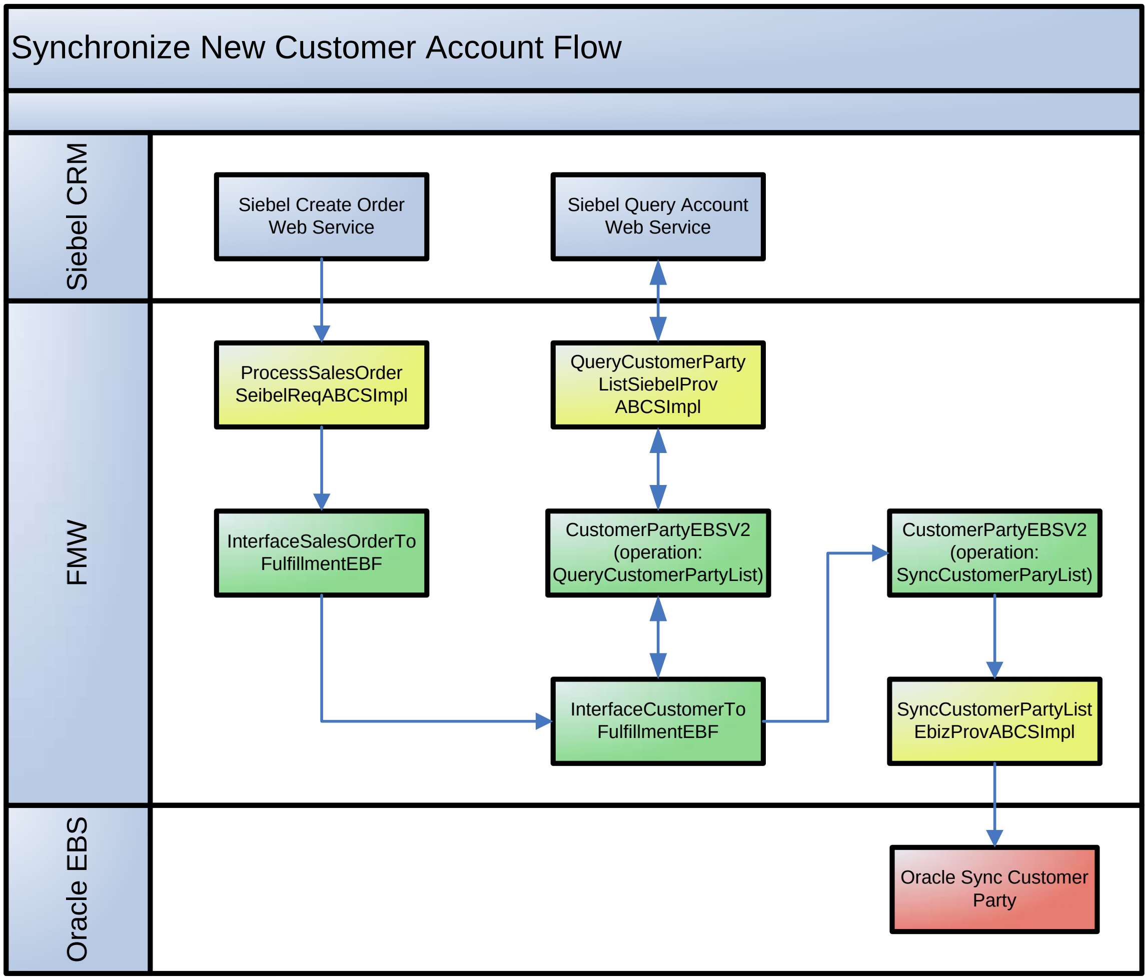
Synchronizing New Customer Accounts from Siebel CRM to Oracle EBS

You initiate the Synchronize New Customer Accounts flow as part of the Order Management flow from Siebel CRM to Oracle EBS. 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 a new customer from Siebel CRM to Oracle EBS consider the following:

- You do not need to inform Oracle EBS of any new creations in Siebel except when attempting to book an order in Oracle Order Management.
- 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 API requires a Site ID and Customer ID in order to tie to existing records in Oracle EBS; otherwise, you can create a new (or duplicate) customer in Oracle EBS.
    - You can also create account contacts and site contacts in Oracle EBS using the Oracle Order Management interface.
- You create Location, Party Site, Account Site, and Site Uses in Oracle EBS as part of the Customer process integration.

This diagram illustrates the Synchronize New Customer Account integration flow:


Synchronize New Customer Account Flow
Siebel CRM
FMW
Oracle EBS
Siebel Create Order Web Service
Siebel Query Account Web Service
ProcessSalesOrder SeibelReqABCSImpl
QueryCustomerParty ListSiebelProv ABCSImpl
InterfaceSalesOrderTo FulfillmentEBF
CustomerPartyEBSV2 (operation: QueryCustomerPartyList)
CustomerPartyEBSV2 (operation: SyncCustomerParyList)
InterfaceCustomerTo FulfillmentEBF
SyncCustomerPartyList EbizProvABCSImpl
Oracle Sync Customer Party

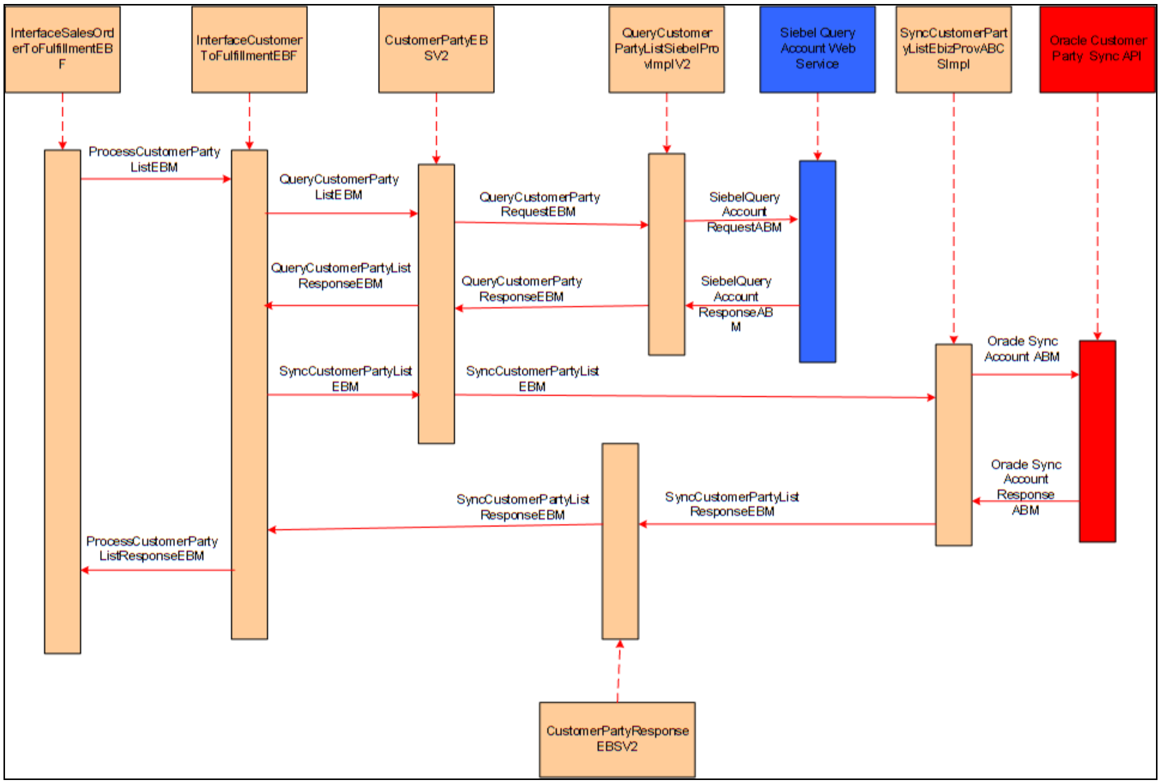
Synchronize New Customer Account integration flow

Synchronize New Customer Account Integration Flow

This integration flow uses the following interfaces:

- InterfaceCustomerToFulfillmentEBF
- CustomerPartyEBSV2
- CustomerPartyResponseEBSV2
- QueryCustomerPartyListSiebelProvABCSImplV2
- SyncCustomerPartyListEbizProvABCSImpl
- SyncCustomerPartyListEbizAdapter

This sequence diagram illustrates the Synchronize Customer Account integration flow:


InterfaceSalesOrderToFulfillmentEBF
InterfaceCustomerToFulfillmentEBF
CustomerPartyEBSV2
QueryCustomerPartyListSiebelProvImplV2
Siebel Query Account Web Service
SyncCustomerPartyListEbizProvABCSImpl
Oracle Customer Party Sync API
ProcessCustomerParty ListEBM
QueryCustomerParty ListEBM
QueryCustomerParty RequestEBM
SiebelQuery Account RequestABM
QueryCustomerPartyList ResponseEBM
QueryCustomerParty ResponseEBM
SiebelQuery Account ResponseABM
Oracle Sync Account ABM
SyncCustomerPartyList EBM
SyncCustomerPartyList EBM
Oracle Sync Account Response ABM
SyncCustomerPartyList ResponseEBM
SyncCustomerPartyList ResponseEBM
ProcessCustomerParty ListResponseEBM
CustomerPartyResponse EBSV2

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 CustomerPartyEBSV2 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/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.

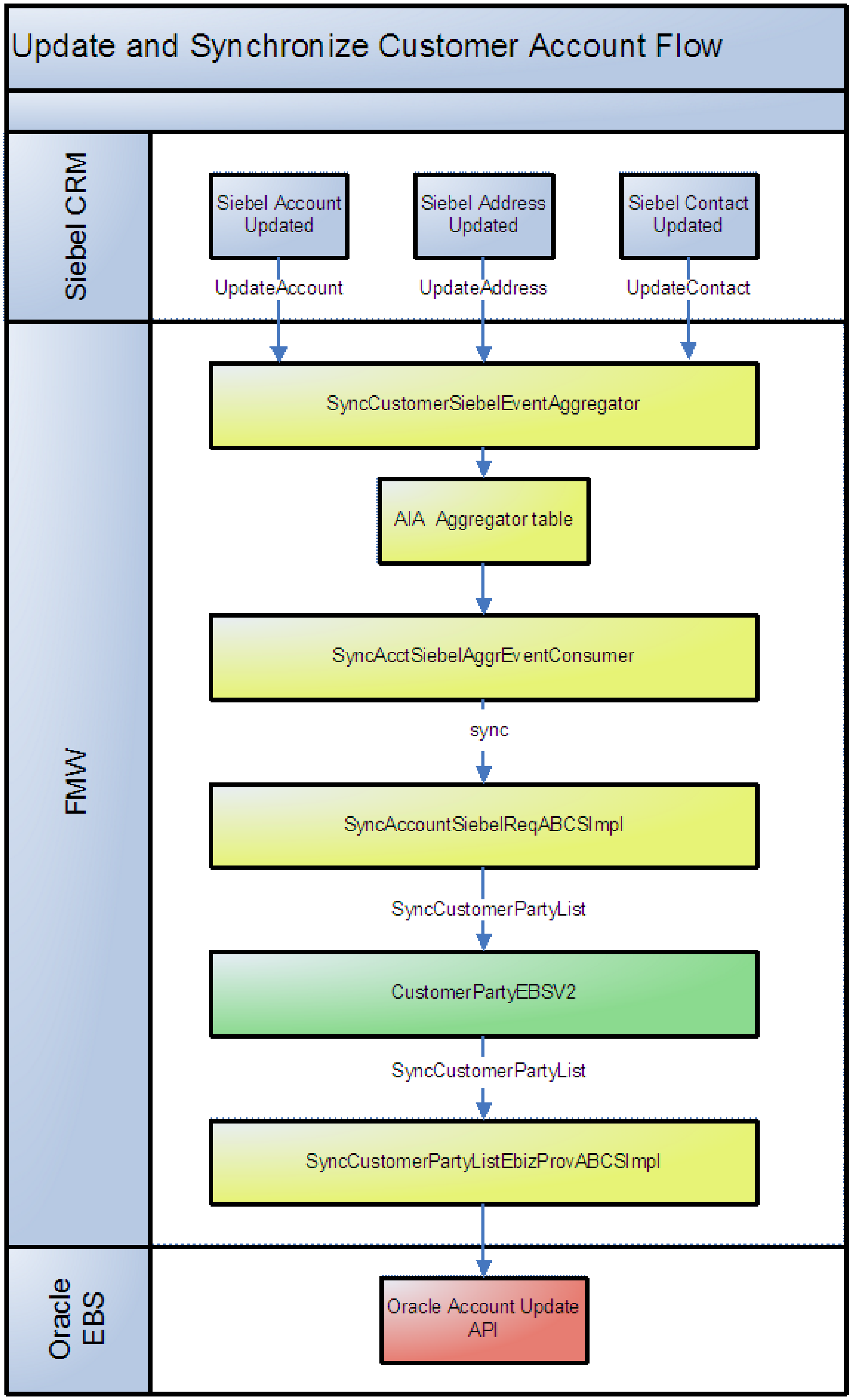
Updating and Synchronizing Customer Accounts from Siebel CRM to Oracle EBS

You initiate the Update and Synchronize Customer Account integration flow in Siebel CRM when you update a customer account. This change must be propagated to Oracle EBS.

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 will not be synchronized to Oracle EBS.

Updates include information such as existing Siebel accounts and changes to ship-to addresses, bill-to addresses, contact points, account/contact profiles, and so on.

This diagram illustrates the Update and Synchronize Customer Account integration flow:



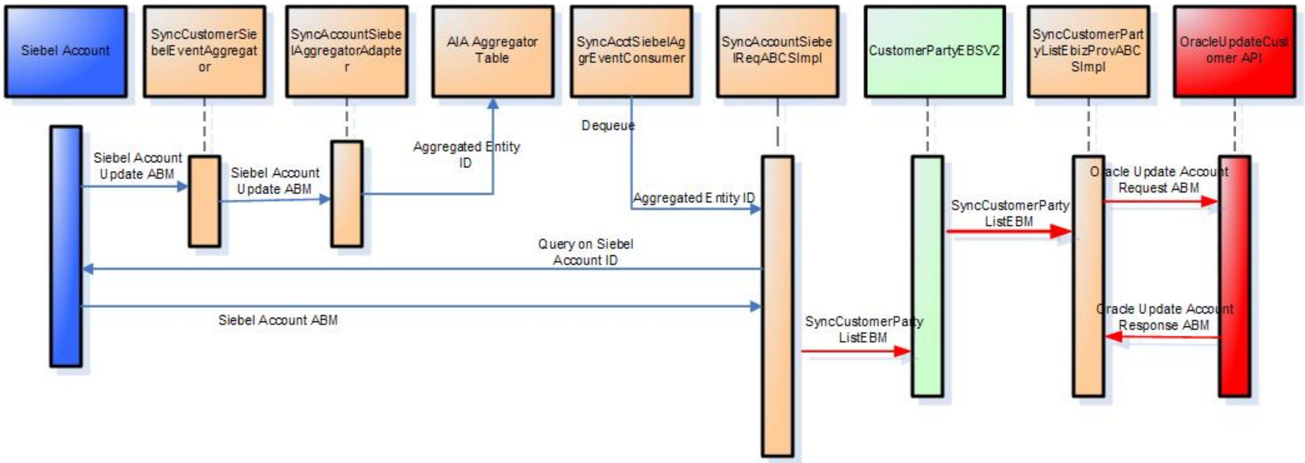
Update and Synchronize Customer Account integration flow

Update and Synchronize Customer Account Integration Flow

This integration flow uses the following interfaces:

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

This sequence diagram illustrates the Update and Synchronize integration flow:


Siebel Account
SyncCustomerSiebelEventAggregator
SyncAccountSiebelAggregatorAdapter
AIA Aggregator Table
SyncAcctSiebelAggrEventConsumer
SyncAccountSiebelReqABCSImpl
CustomerPartyEBSV2
SyncCustomerPartyListEbizProvABCSImpl
OracleUpdateCustomer API
Siebel Account Update ABM
Siebel Account Update ABM
Aggregated Entity ID
Dequeue
Aggregated Entity ID
Query on Siebel Account ID
Siebel Account ABM
SyncCustomerParty ListEBM
SyncCustomerParty ListEBM
Oracle Update Account Request ABM
Oracle Update Account Response ABM

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 one of the aggregator adapters (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 invokes the ‘sync’ operation of the Siebel requester ABC implementation service.

5. The Siebel Requester ABC implementation, SyncAccountSiebelReqABCSImpl, queries the Siebel application back 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.

Updating Polling Interval on FMW 10.1.3.4 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 usecases:

1. Go to the Oracle ESB Console.
2. Look for the SyncAcctSiebelAggrEventConsumer service under AIASystem > Siebel > ABCS.
3. Select this service and go to the Properties tab on the right.

   Click on the '+' button on the Properties table. This displays the property 'PollingInterval' set to 90. Change this value to the necessary value required by the user scenario.

4. Click Apply to save the changes.
5. Restart the SOA server for the changes to take effect.

   Perform the same steps as for the contact consumer service, SyncContSiebelAggrEventConsumer, if needed.

Synchronizing Customer Accounts from Oracle EBS to Siebel CRM

You initiate the Synchronize Customer Account integration flow in Oracle EBS when you create a new customer account or update an existing account. Details related to the account (account number, addresses, contacts, phone number, fax number, email address, web address) are reflected in the target system, Siebel CRM.

When an account is created or updated in Oracle EBS, a record is written to a tracking table. By raising a Trading Community Architecture (TCA) business event, all of the account records in the tracking table are stamped with an event identifier and enqueue event message to AQ. The synchronize account process will dequeue from AQ, read the event identifier, and invoke the TCA API to extract all of the created or updated account information.

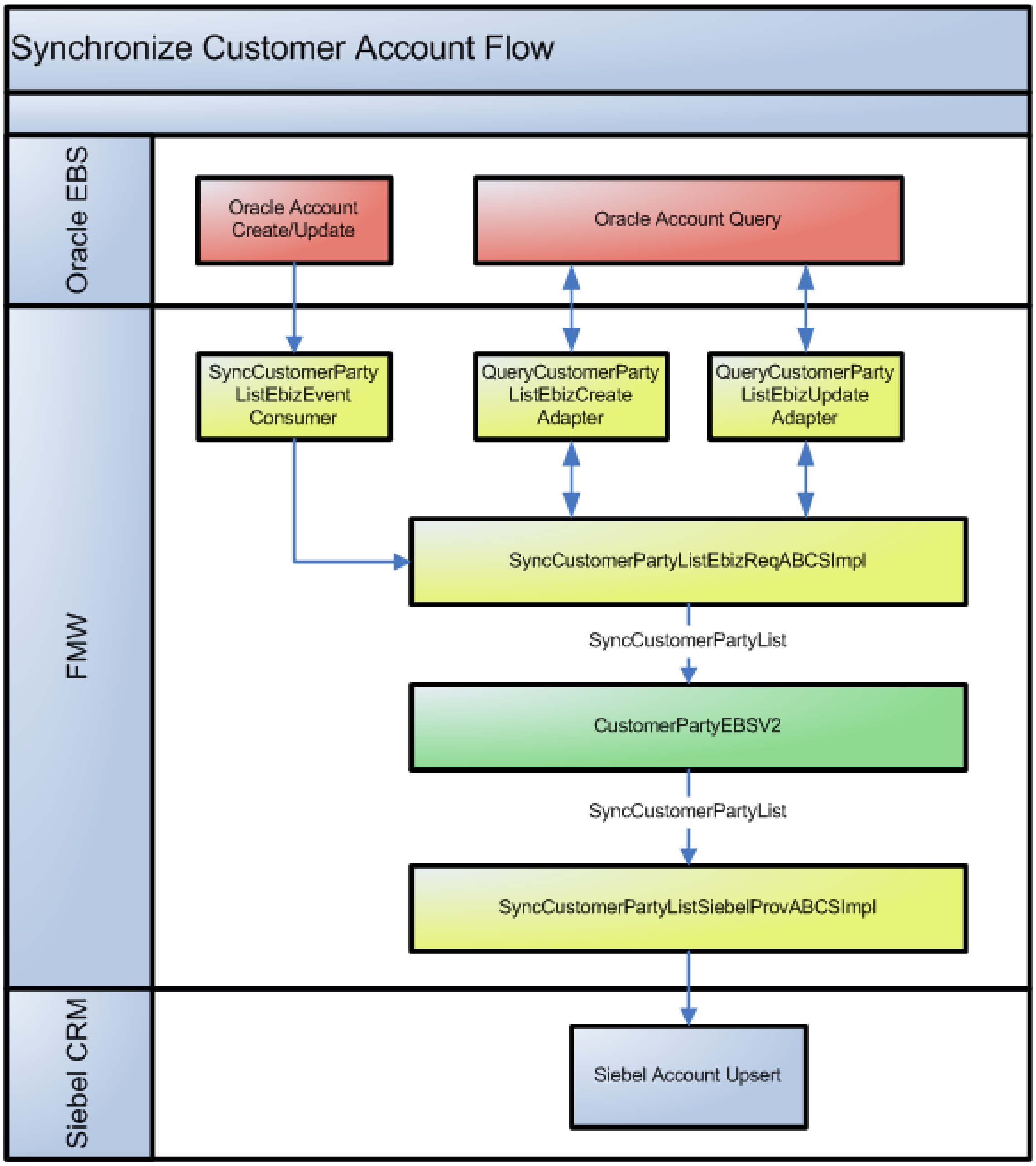
A create or update event is tracked:

- 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.

This diagram illustrates the Synchronize Customer Account integration flow:


Synchronize Customer Account Flow
Oracle EBS
FMW
Siebel CRM
Oracle Account Create/Update
Oracle Account Query
SyncCustomerParty ListEbizEvent Consumer
QueryCustomerParty ListEbizCreate Adapter
QueryCustomerParty ListEbizUpdate Adapter
SyncCustomerPartyListEbizReqABCSImpl
SyncCustomerPartyList
CustomerPartyEBSV2
SyncCustomerPartyList
SyncCustomerPartyListSiebelProvABCSImpl
Siebel Account Upsert

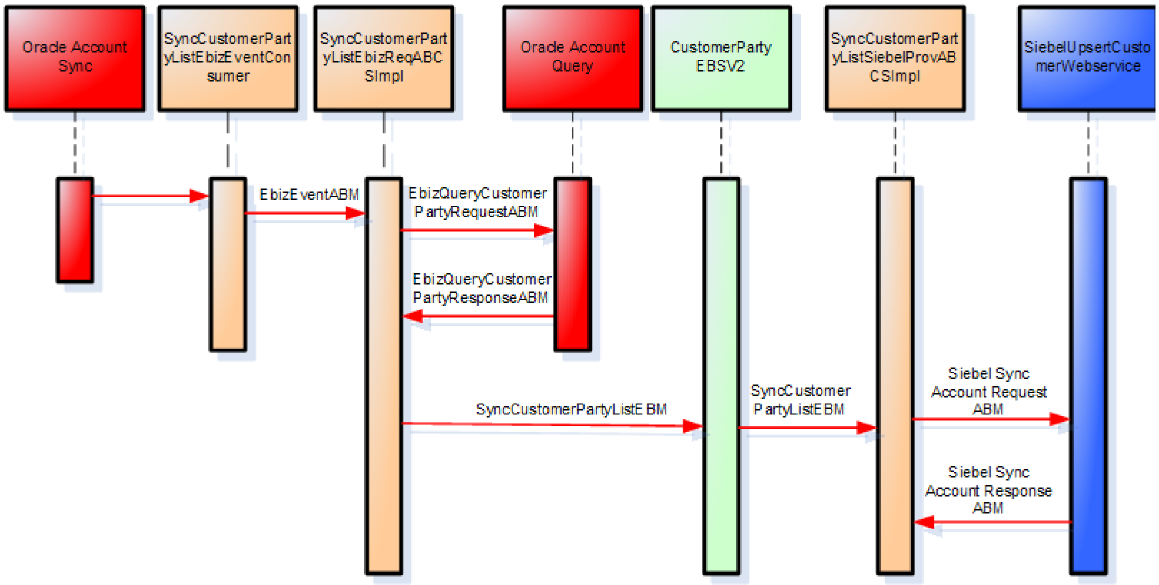
Synchronize Customer Account integration flow

Synchronize Customer Account Integration Flow

This integration flow uses the following interfaces:

- SyncCustomerPartyListEbizEventConsumer
- QueryCustomerPartyListEbizCreateAdapter
- QueryCustomerPartyListEbizUpdateAdapter
- SyncCustomerPartyListEbizReqABCSImpl
- CustomerPartyEBSV2
- SyncCustomerPartyListSiebelProvABCSImpl

This sequence diagram illustrates the Synchronize Account integration flow:


Oracle Account Sync
SyncCustomerPartyListEbizEventConsumer
SyncCustomerPartyListEbizReqABCSImpl
Oracle Account Query
CustomerParty EBSV2
SyncCustomerPartyListSiebelProvABCSImpl
SiebelUpsertCustomerWebservice
EbizEventABM
EbizQueryCustomerPartyRequestABM
EbizQueryCustomerPartyResponseABM
SyncCustomerPartyListEBM
SyncCustomerPartyListEBM
Siebel Sync Account Request ABM
Siebel Sync Account Response ABM

Synchronize 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 SyncCustomerPartyListEbizEventConsumer is listening. Different events are raised for a create scenario and an update scenario.

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

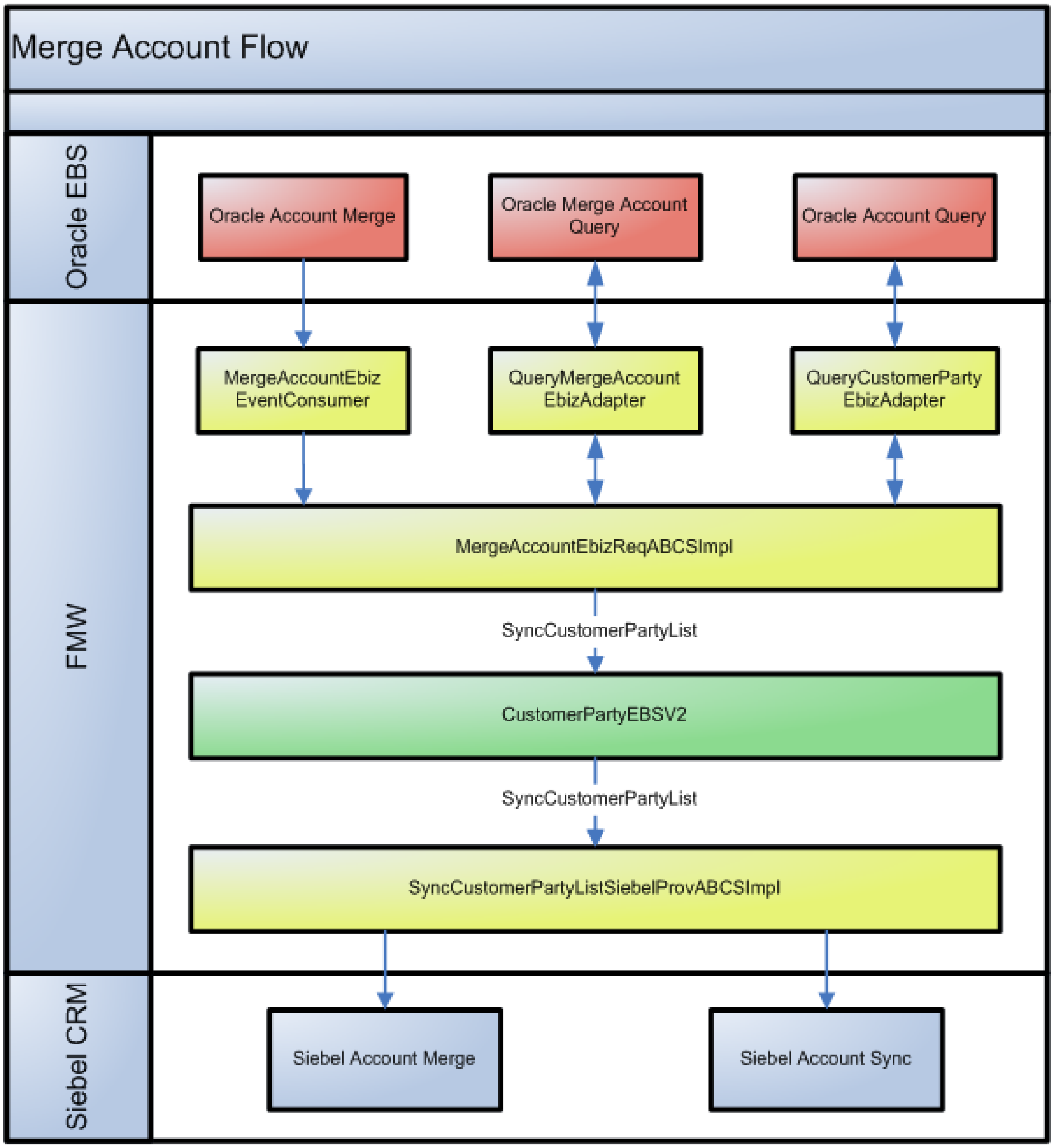
   The Account Update flow ends at this point; no response is sent back to the EBS.

Merging Accounts from Oracle EBS to Siebel CRM

You initiate the Merge Account integration flow in Oracle EBS when you merge an account 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.

This diagram illustrates the Merge Account integration flow:


Merge Account Flow
Oracle EBS
FMW
Siebel CRM
Oracle Account Merge
Oracle Merge Account Query
Oracle Account Query
MergeAccountEbiz EventConsumer
QueryMergeAccount EbizAdapter
QueryCustomerParty EbizAdapter
MergeAccountEbizReqABCSImpl
SyncCustomerPartyList
CustomerPartyEBSV2
SyncCustomerPartyList
SyncCustomerPartyListSiebelProvABCSImpl
Siebel Account Merge
Siebel Account Sync

Merge Account integration flow

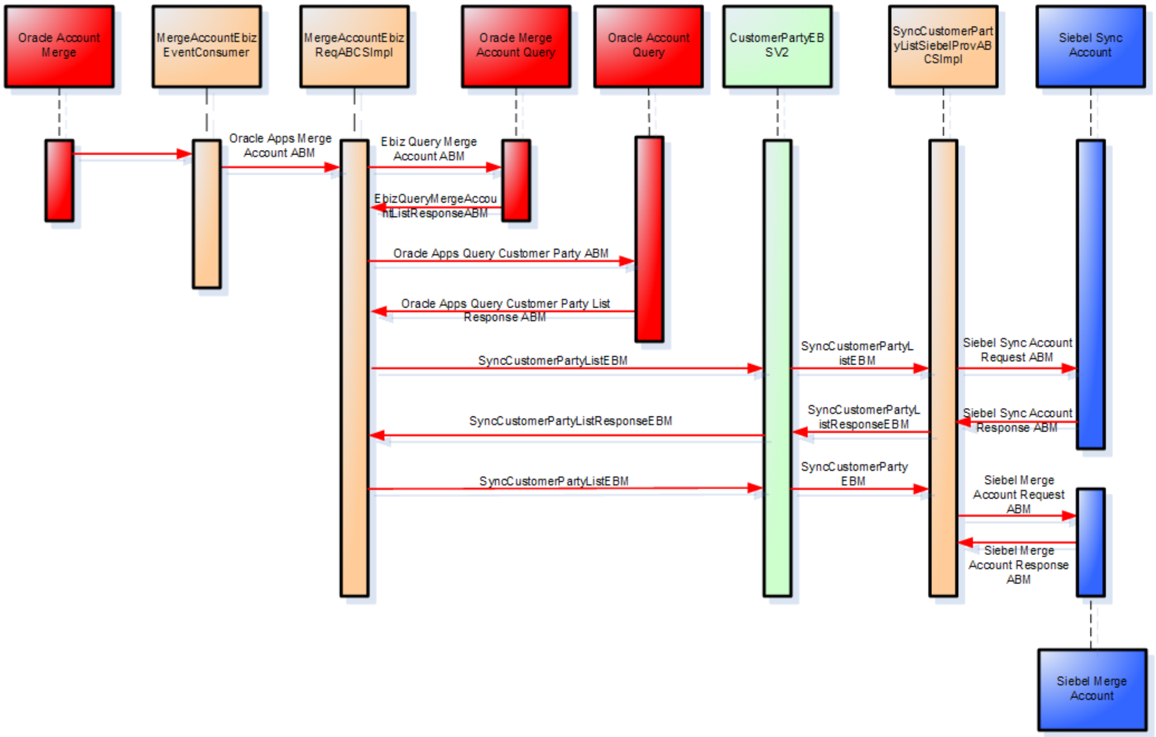
Merge Account Integration Flow

This integration flow uses the following interfaces:

- MergeAccountEbizEventConsumer
- MergeAccountEbizReqABCSImpl
- CustomerPartyEBSV2

- CustomerPartyResponseEBSV2
- SyncCustomerPartySiebelProvABCSImpl
- QueryMergeAccountEbizAdapter
- QueryCustomerPartyEbizAdapter

This sequence diagram illustrates the Merge Account integration flow:


Oracle Account Merge
MergeAccountEbizEventConsumer
MergeAccountEbizReqABCSImpl
Oracle Merge Account Query
Oracle Account Query
CustomerPartyEBSV2
SyncCustomerPartyListSiebelProvABCSImpl
Siebel Sync Account
Oracle Apps Merge Account ABM
Ebiz Query Merge Account ABM
EbizQueryMergeAccountListResponseABM
Oracle Apps Query Customer Party ABM
Oracle Apps Query Customer Party List Response ABM
SyncCustomerPartyListEBM
SyncCustomerPartyListEBM
Siebel Sync Account Request ABM
SyncCustomerPartyListResponseEBM
SyncCustomerPartyListResponseEBM
Siebel Sync Account Response ABM
SyncCustomerPartyListEBM
SyncCustomerParty EBM
Siebel Merge Account Request ABM
Siebel Merge Account Response ABM
Siebel Merge Account

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 selected during the Merge to create the same address or not) 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 being transformed to the SyncCustomerPartyListResponseEBM.

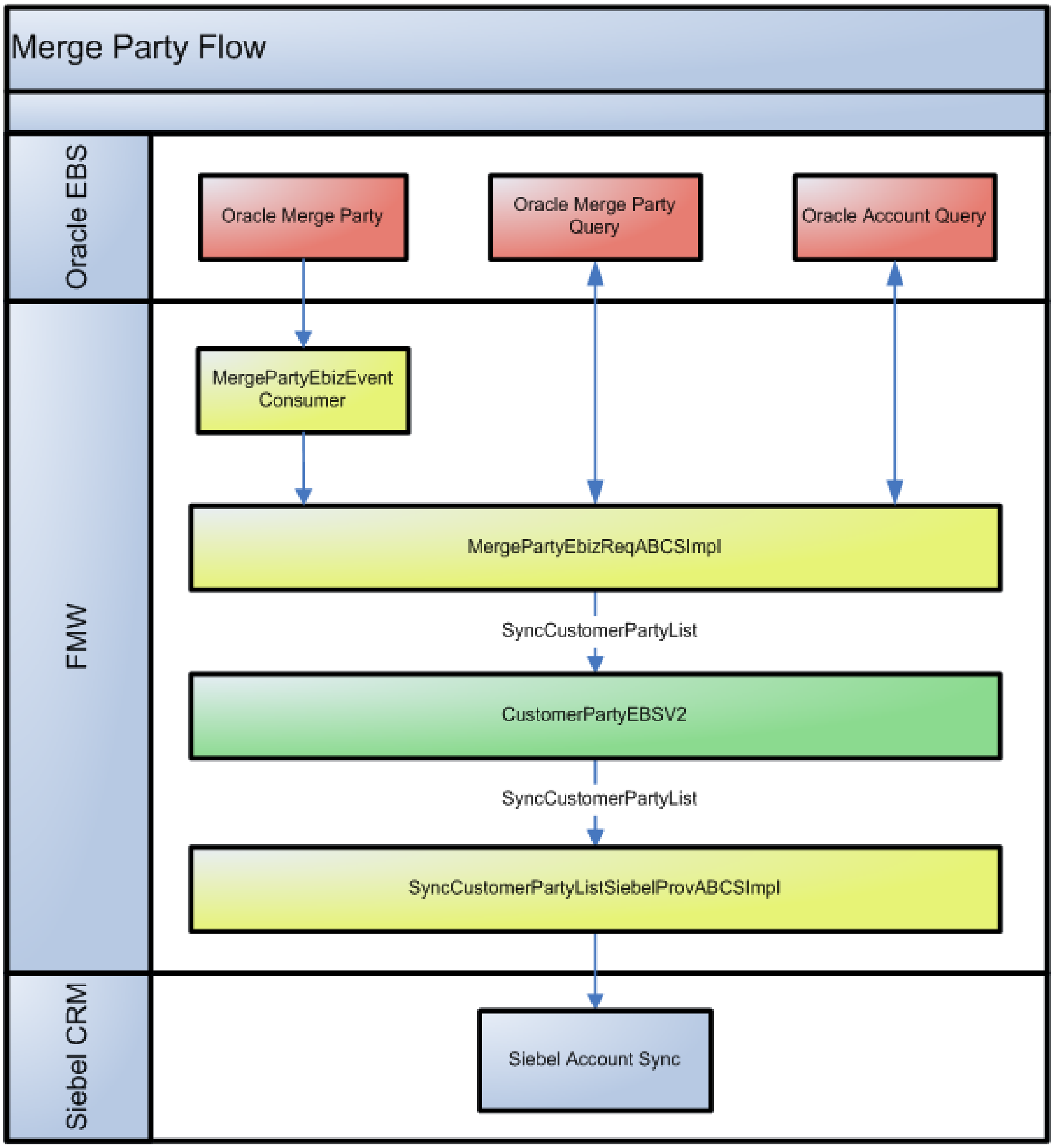
11. The Merge Account flow ends with no further responses sent back to the EBS.

Merging Parties from Oracle EBS to Siebel CRM

You initiate the Merge Party integration flow in Oracle EBS when you merge an organization party into another party in Oracle EBS. Since 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.

This diagram illustrates the Merge Party integration flow:



Merge Party integration flow

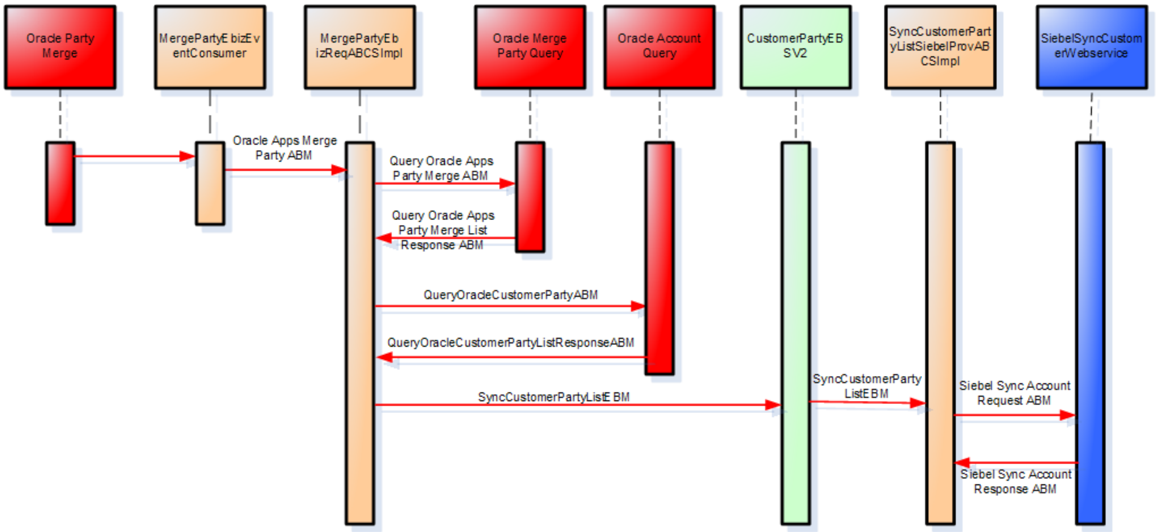
Merge Party Integration Flow

This integration flow uses the following interfaces:

- MergePartyEbizEventConsumer
- MergePartyEbizReqABCSImpl
- CustomerPartyEBSV2

- SyncCustomerPartySiebelProvABCSImpl
- QueryRelatedOrgCustEbizAdapter
- QueryMergeOrgCustEbizAdapter
- QueryPartyMergeEbizAdapter

This sequence diagram illustrates the Merge Party integration flow:


Oracle Party Merge
MergePartyEbizEventConsumer
MergePartyEbizReqABCSImpl
Oracle Merge Party Query
Oracle Account Query
CustomerPartyEBSV2
SyncCustomerPartyListSiebelProvABCSImpl
SiebelSyncCustomerWebservice
Oracle Apps Merge Party ABM
Query Oracle Apps Party Merge ABM
Query Oracle Apps Party Merge List Response ABM
QueryOracleCustomerPartyABM
QueryOracleCustomerPartyListResponseABM
SyncCustomerPartyListEBM
SyncCustomerParty ListEBM
Siebel Sync Account Request ABM
Siebel Sync Account Response ABM

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.

   Since 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 sent back to EBS.

Data Requirements

Data requirements indicate the mandatory data that must be provided to make the flow successful:

- For customer account synchronizations, the Siebel Organization used must be mapped to an Oracle EBS Operating Unit.
- Oracle EBS profile options must be specified.
- The contact associated with the order must have the same account associated to it.
- Contact Name must be specified on the Payments tab of the Sales Order screen.
- The address must have Address Line 1, City, State, and Zip Code.
- Locations cannot be shared by different organization parties in Oracle EBS, though addresses can be shared between accounts in Siebel. Thus, the address should be recreated in Siebel.
- In Oracle EBS, the contact should be associated at the account level only.
  For the Order Management API to process the contact, it should not be associated at multiple levels of an account. For example, not at the account site (address) level.
- If the contact used in Siebel CRM during Order submit is associated with more than one account and contact points (phone numbers or email address) are present, then Order submit will not be supported since the Oracle EBS API for contact update/create does not support shared contact points.

For more information about setting Siebel Organizations and about Oracle EBS Operating Units and profile options, see Chapter 13: Implementing the Order to Cash: Siebel CRM – EBS Process Integration Pack

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.

The following is the list of 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 the Siebel CRM Integration Pack for Oracle Order Management Addendum

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 EBS web services, see the following Oracle E-Business Suite references:
E-Business Suite Electronic Technical Reference Manual (eTRM) located on My Oracle Support under the E-Business Suite Information Center,
Oracle Integration Repository located at http://irep.oracle.com, Oracle Applications Release 11.5.10+ Online Documentation Library, located on the Oracle Technology Network (http://www.oracle.com/technology/documentation/applications.html)

Core AIA Components

The Customer Management integration uses the following horizontal components:

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

The core EBO and EBM XSD files can be located by EBO within this parent folder: http://[HOST:PORT]/AIAComponents/EnterpriseObjectLibrary/Core/EBO/.

The core EBS WSDL files can be located by EBO within this parent folder: http://[HOST:PORT]/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/.

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

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

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

For more information, see *Oracle Application Integration Architecture - Foundation Pack: Integration Developer’s Guide*, “Extensibility for Oracle AIA Artifacts,” Extending EBOs

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
- SyncCustomerPartyListEbizEventConsumer
- MergeAccountEbizEventConsumer
- MergePartyEbizEventConsumer
- SyncCustomerPartyListEbizAdapter

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

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

CustomerPartyEBSV2

The CustomerPartyEBSV2 Enterprise Business Service is an ESB 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 enterprise business flow, see Oracle Application Integration Architecture – Foundation Pack: Integration Developer's Guide, "Designing and Developing Enterprise Business Services" and Oracle Application Integration Architecture - Foundation Pack: Concepts and Technologies Guide, "Understanding Enterprise Business Services"

CustomerPartyResponseEBSV2

The CustomerPartyResponseEBSV2 Enterprise Business Service is an ESB 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 enterprise business service.

The CustomerPartyResponseEBSV2 enterprise business service uses this operation: SyncCustomerPartyListResponse.

For more information about this enterprise business flow, see *Oracle Application Integration Architecture - Foundation Pack: Integration Developer's Guide*, "Designing and Developing Enterprise Business Services" and *Oracle Application Integration Architecture - Foundation Pack: Concepts and Technologies Guide*, "Understanding Enterprise Business Services"

InterfaceCustomerToFulfillmentEBF

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

This enterprise business flow creates or synchronizes all the customer accounts and related components (like address and contacts) in an order 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 sent back 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 already will have been identified within the Order flow, and this information is passed to this service through the EBM header. The source CRM system must also be passed 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 source CRM system, the processes error out.

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.

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.

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 will have one synchronous request/reply operation: QueryCustomerPartyList.

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

SyncCustomerPartyListSiebelProvABCSImpl

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

- Creating or updating customer account in Oracle Apps. This service is initiated to synchronize customer account from Oracle to Siebel. An update web service call to Siebel is invoked.
- Before merging an account, this service is initiated to synchronize surviving account information from Oracle to Siebel. In the sync operation, if there is an Address with Row ID 100 in Siebel but not in request message, then the Address will be removed. A synchronize web service call to Siebel will be invoked.

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

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

SyncCustomerPartyListEbizReqABCSImpl

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

This service has two asynchronous operations: create and update.

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.

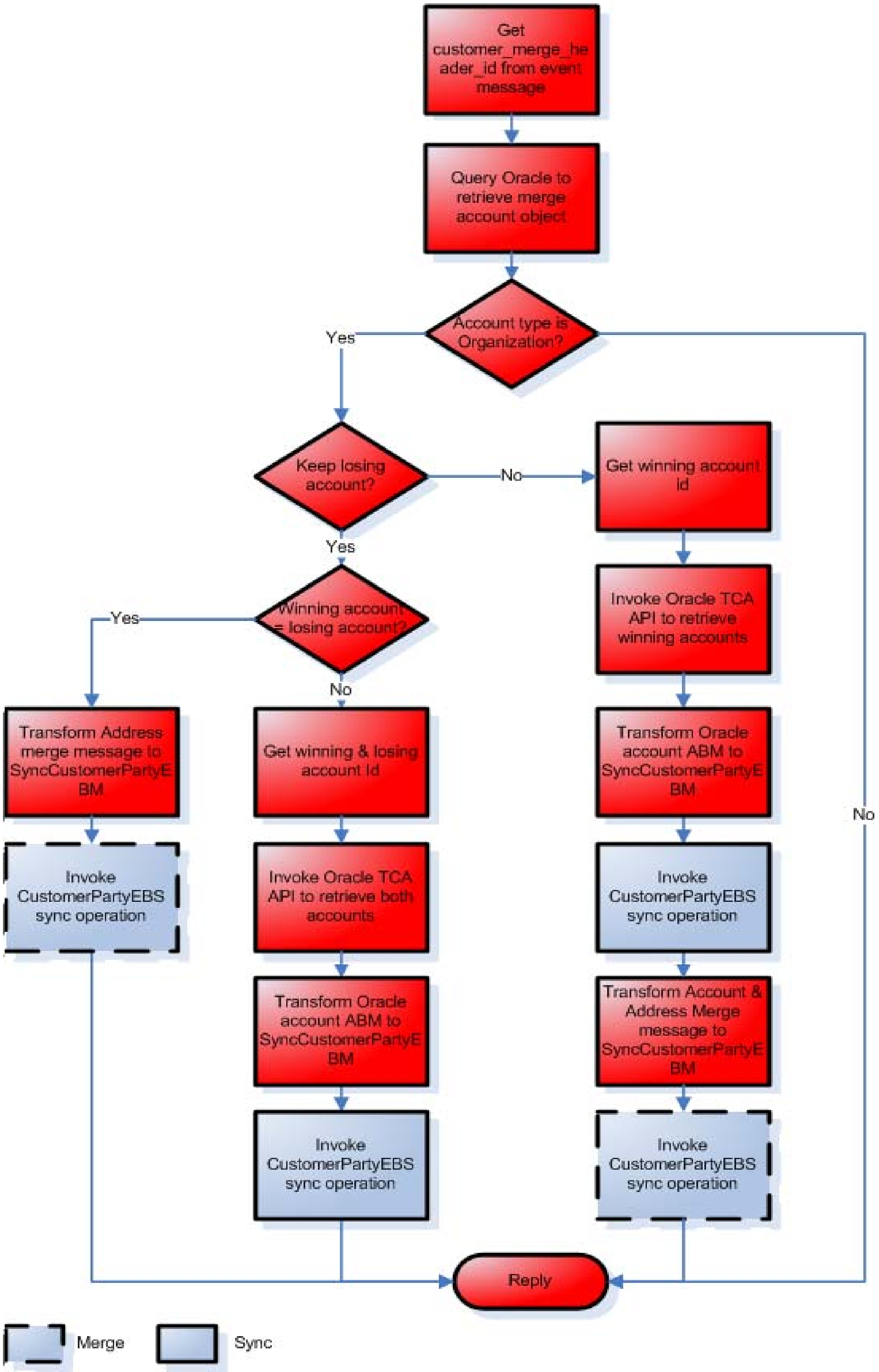
This service has one synchronous request/reply operation: SyncCustomerPartyList.

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.

This diagram illustrates the MergeAccountEbizReqABCSImpl process activity:


Get customer_merge_header_id from event message
Query Oracle to retrieve merge account object
Account type is Organization?
Yes
No
Keep losing account?
No
Get winning account id
Yes
Winning account = losing account?
Yes
No
Invoke Oracle TCA API to retrieve winning accounts
Transform Address merge message to SyncCustomerPartyEBM
Get winning & losing account Id
Transform Oracle account ABM to SyncCustomerPartyEBM
Invoke CustomerPartyEBS sync operation
Invoke Oracle TCA API to retrieve both accounts
Invoke CustomerPartyEBS sync operation
Transform Oracle account ABM to SyncCustomerPartyEBM
Transform Account & Address Merge message to SyncCustomerPartyEBM
Invoke CustomerPartyEBS sync operation
Invoke CustomerPartyEBS sync operation
Reply
Merge
Sync

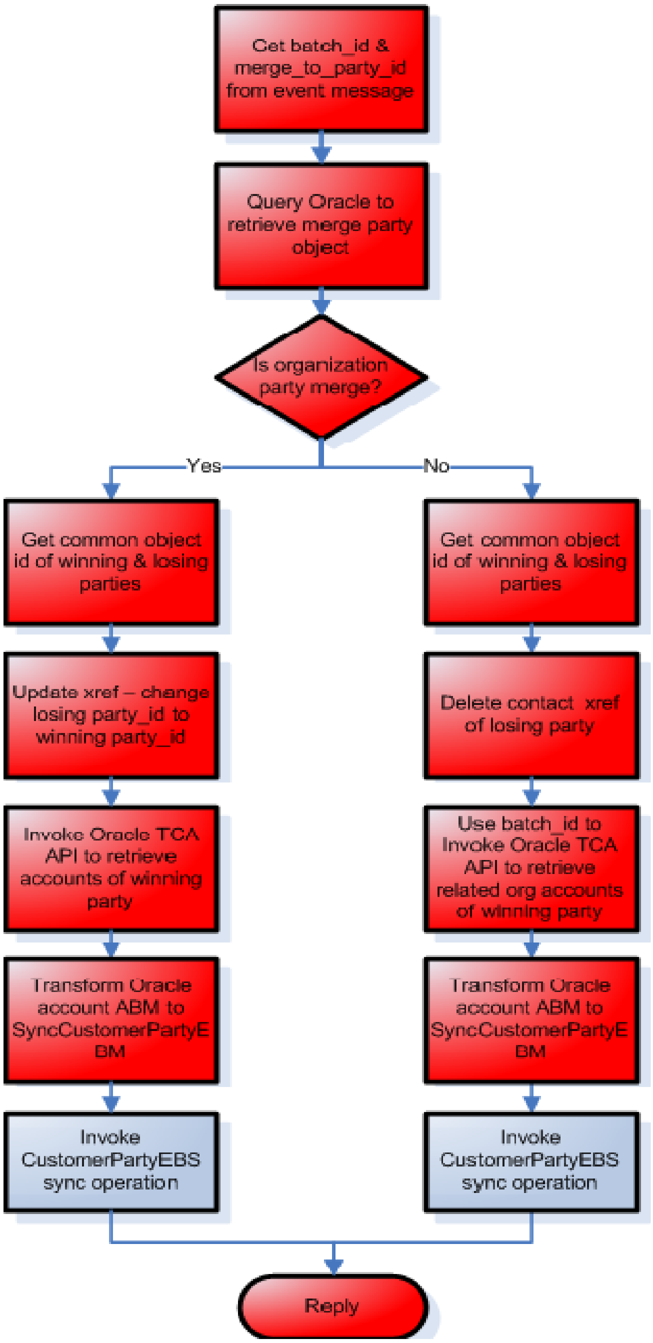
MergeAccountEbizReqABCSImpl process activity diagram

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.

This diagram illustrates the MergePartyEbizReqABCSImpl process activity:


Get batch_id & merge_to_party_id from event message
Query Oracle to retrieve merge party object
Is organization party merge?
Yes
No
Get common object id of winning & losing parties
Update xref – change losing party_id to winning party_id
Invoke Oracle TCA API to retrieve accounts of winning party
Transform Oracle account ABM to SyncCustomerPartyEBM
Invoke CustomerPartyEBS sync operation
Get common object id of winning & losing parties
Delete contact xref of losing party
Use batch_id to Invoke Oracle TCA API to retrieve related org accounts of winning party
Transform Oracle account ABM to SyncCustomerPartyEBM
Invoke CustomerPartyEBS sync operation
Reply

MergePartyEbizReqABCSImpl process activity diagram

SyncCustomerSiebelEventAggregator

The SyncCustomerSiebelEventAggregator is implemented as a BPEL process since it involves JMS Header manipulations that cannot be done in ESB. 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 programming model, see *Oracle Application Integration Architecture – Foundation Pack: Integration Developer's Guide,* "Describing the Event Aggregation Programming Model"

SyncAccountSiebelAggregatorAdapter

The SyncAccountSiebelAggregatorAdapter is implemented as an ESB 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 programming model, see *Oracle Application Integration Architecture – Foundation Pack: Integration Developer's Guide,* "Describing the Event Aggregation Programming Model"

SyncContactSiebelAggregatorAdapter

The SyncContactSiebelAggregatorAdapter is implemented as an ESB 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 programming model, see *Oracle Application Integration Architecture – Foundation Pack: Integration Developer's Guide,* "Describing the Event Aggregation Programming Model"

SyncAddressSiebelAggregatorAdapter

The SyncAddressSiebelAggregatorAdapter is implemented as an ESB 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 programming model, see *Oracle Application Integration Architecture – Foundation Pack: Integration Developer’s Guide,* “Describing the Event Aggregation Programming Model”

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 ABCS, SyncAccountSiebelReqABCSImpl.

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

SyncCustomerPartyListEbizEventConsumer

The SyncCustomerPartyListEbizEventConsumer 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, correlation ID is oracle.apps.ar.hz.OrgCustBO.create. For customer update, 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

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.

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.

SyncCustomerPartyListEbizAdapter

The SyncCustomerPartyListEbizAdapter service is an Oracle EBS Adapter service registered in ESB. 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, and it is invoked by the SyncCustomerPartyListEbizProvABCSImpl 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. By registering this adapter service in ESB, ESB will expose a SOAP binding, which will be used in this integration to invoke the service from the SyncCustomerPartyListEbizProvABCSImpl.

Chapter 4: Understanding Process Integration for Product Management

This chapter discusses:

- Process integration for Product Management
- Item synchronization
- Bill of material (BOM) synchronization
- Data requirements
- Siebel Customer Relationship Management (CRM) interfaces
- Oracle E-Business Suite (Oracle EBS) interfaces
- Core Oracle Application Integration Architecture (AIA) components
- Integration services

Process Integration for Product Management

The Process Integration for Product Management supports the following integration flows:

- **Synchronize Items** 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** 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** enables the extract, transformation, and load 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: Describing the Initial Data Loads

Prerequisites

The Process Integration for Product Management has no prerequisites.

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 will not be 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 between Oracle EBS and Siebel CRM after the initial load processes are performed:

    a All of the BOM components (children) and the BOM product (parent) are available in Siebel CRM prior to creating the relationship using the BOM synchronization process.

    b Non-orderable items that are part of the product BOM structure will be synchronized as part of the BOM synchronization process.

    c Although the relationships between the parent and children will be synchronized, the characteristics of the child components, such as Optional, Mandatory, SubAssembly, and so forth, will not be synchronized or maintained in Siebel CRM.

4. In Siebel CRM, there is no mechanism to identify which products have a sales BOM structure in Oracle EBS.

    The Siebel product administrator must be aware that a given product has a specific structure in Oracle EBS so that she can make an appropriate request to have the structure synchronized into Siebel CRM.

5. The following association exists between the BOM ITEM TYPE item attribute in Oracle EBS and the STRUCTURE TYPE product attribute in Siebel CRM for the purpose of synchronizing Bill of Material structures into Siebel and making them configurable and customizable:

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 will not be able to make request to synchronize Bill of Material structures for any items in Oracle EBS that have been set up with a BOM ITEM TYPE of “Standard”.

    Since 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:

    a Item Name (part of the Key Flex Field structure)

    b Item Description

    c Item Type

    d Item Status

    e Primary UOM

f Order Management related attributes (orderable flag, customer ordered flag)

g 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, ATO models that is child components of PTO models.

9. BOM synchronization must be manually invoked 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 has structures that also need to be imported into Siebel, then the Product Administrator in Siebel will do a manual Request for BOM from the Product Admin screens.

11. Non-orderable items are not loaded into Siebel as simple products; however, non-orderable items that are part of a BOM structure are loaded into Siebel as component products when a BOM is requested and synchronized to Siebel.

12. The organizational setup on the Siebel side is a critical element in this process integration.

For more information about the organization setup, see Chapter 13: Implementing the Order to Cash: Siebel CRM – EBS Process Integration Pack

13. For synchronizing simple and complex products into Siebel CRM, assume that Siebel can fully support duplicate product names across, but not within, organizations.

    The organization must be unique.

14. Item Name in Oracle EBS is comprised of concatenated key flex field segments.

    It cannot exceed 50 characters in length.

15. Special characters, such as & should not be used in the Oracle EBS Item Name definition.

16. If an existing Item Name in Oracle EBS is updated and synchronized to Siebel, a new product will be created in Siebel.

17. A known and identified interdependency exists between PRICE TYPE values and UNIT OF MEASURE values assigned to an item in Siebel.

    Since PRICE TYPE is not an attribute mapped in the integration between Oracle EBS and Siebel, you must set up a “One-Time” Price Type in Siebel, and you must 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.

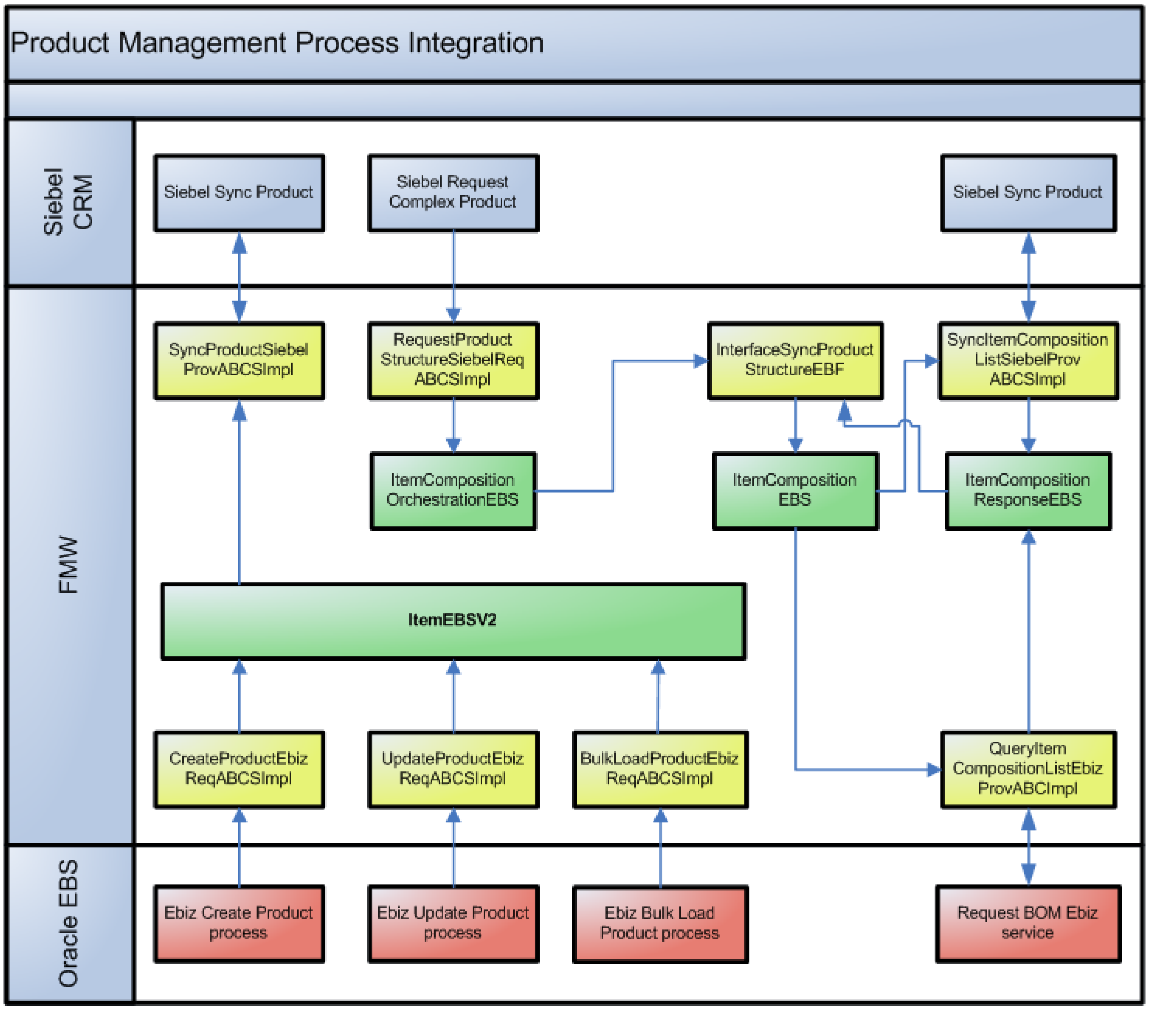
18. Deleting items in Oracle EBS will not result in a request to delete simple products in Siebel because no event mechanisms exist to accomplish this operation.

    Any deletion of simple products in Siebel must be run as a manual process by an administrator in Siebel.

19. Only items associated with Oracle EBS “OE Item Validation Org” is synchronized from Oracle EBS to Siebel CRM.

20. Custom and user-defined attributes for items or BOMs are not supported.

This diagram illustrates the overall flow for the process integration:


Product Management Process Integration
Siebel CRM
FMW
Oracle EBS
Siebel Sync Product
Siebel Request Complex Product
Siebel Sync Product
SyncProductSiebel ProvABCSImpl
RequestProduct StructureSiebelReq ABCSImpl
InterfaceSyncProduct StructureEBF
SyncItemComposition ListSiebelProv ABCSImpl
ItemComposition OrchestrationEBS
ItemComposition EBS
ItemComposition ResponseEBS
ItemEBSV2
CreateProductEbiz ReqABCSImpl
UpdateProductEbiz ReqABCSImpl
BulkLoadProductEbiz ReqABCSImpl
QueryItem CompositionListEbiz ProvABCImpl
Ebiz Create Product process
Ebiz Update Product process
Ebiz Bulk Load Product process
Request BOM Ebiz service

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 provide visibility of saleable products in the 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.
  Once a simple product is synchronized in Siebel, 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, optional, or both components. Once a complex product is synchronized in Siebel, 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, product attributes and BOM structures cannot be changed. However, product administrators can leverage eligibility and compatibility rules in Siebel. 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 includes 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 only synchronizes the individual items in a BOM. The relationships or structures of the BOM must be synchronized as an independent process.

For more information about initial data loads, see Chapter 2: Describing the Initial Data Loads

Subsequent creation or updates to simple products in the Oracle product master trigger a business event in Oracle EBS that synchronizes these products to Siebel in realtime.

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

Item Synchronization

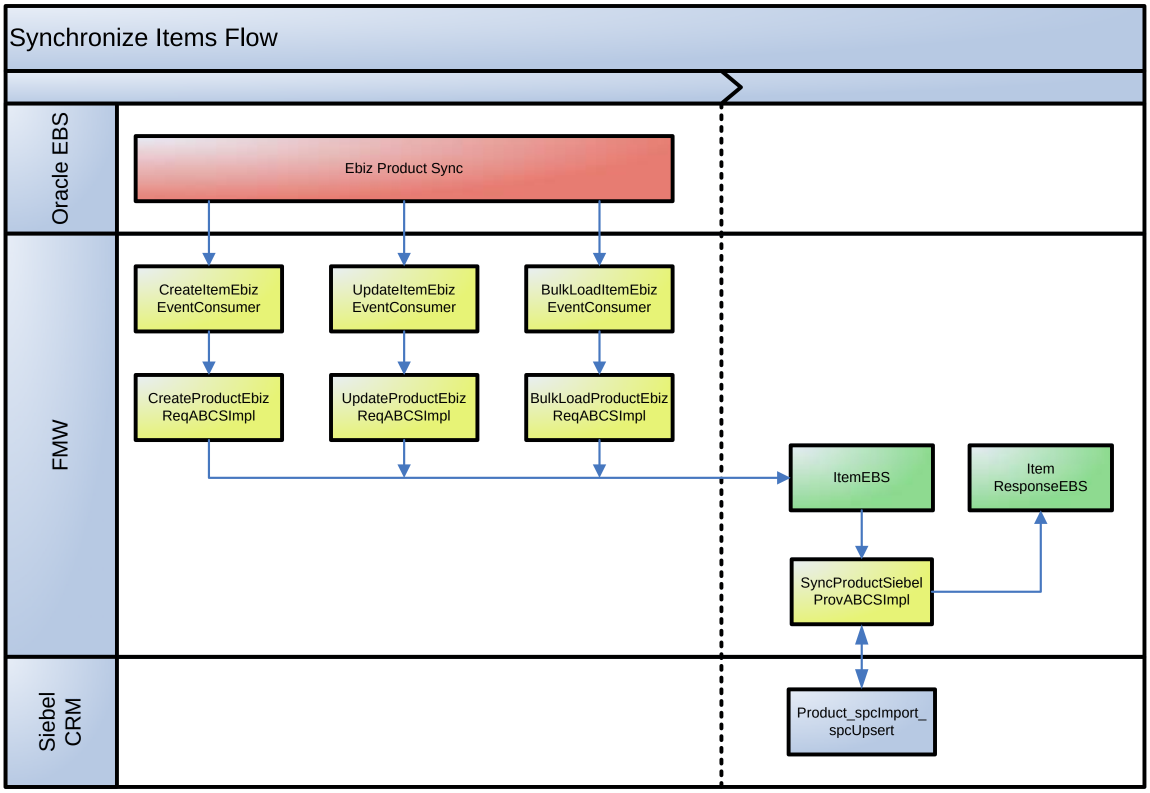
The Synchronize Items integration flow enables the synchronization of items from Oracle EBS to Siebel CRM. This synchronization is a one-way feed from Oracle EBS to CRM. After items are created or updated in Oracle EBS, a business event is triggered that enables the synchronization of items from Oracle EBS to Siebel CRM. This flow is used for incremental loads; high-volume initial loads should use the Initial Data Load integration flow that is supported by this Process Integration Pack (PIP).

For more information about initial data loads, see Chapter 2: Describing the Initial Data Loads

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 unit of measure (UOM)
- Order Management related attributes

This diagram illustrates the flow for synchronizing items between Oracle EBS and Siebel CRM:


Synchronize Items Flow
Oracle EBS
FMW
Siebel CRM
Ebiz Product Sync
CreateItemEbiz EventConsumer
UpdateItemEbiz EventConsumer
BulkLoadItemEbiz EventConsumer
CreateProductEbiz ReqABCSImpl
UpdateProductEbiz ReqABCSImpl
BulkLoadProductEbiz ReqABCSImpl
ItemEBS
Item ResponseEBS
SyncProductSiebel ProvABCSImpl
Product_spcImport_ spcUpsert

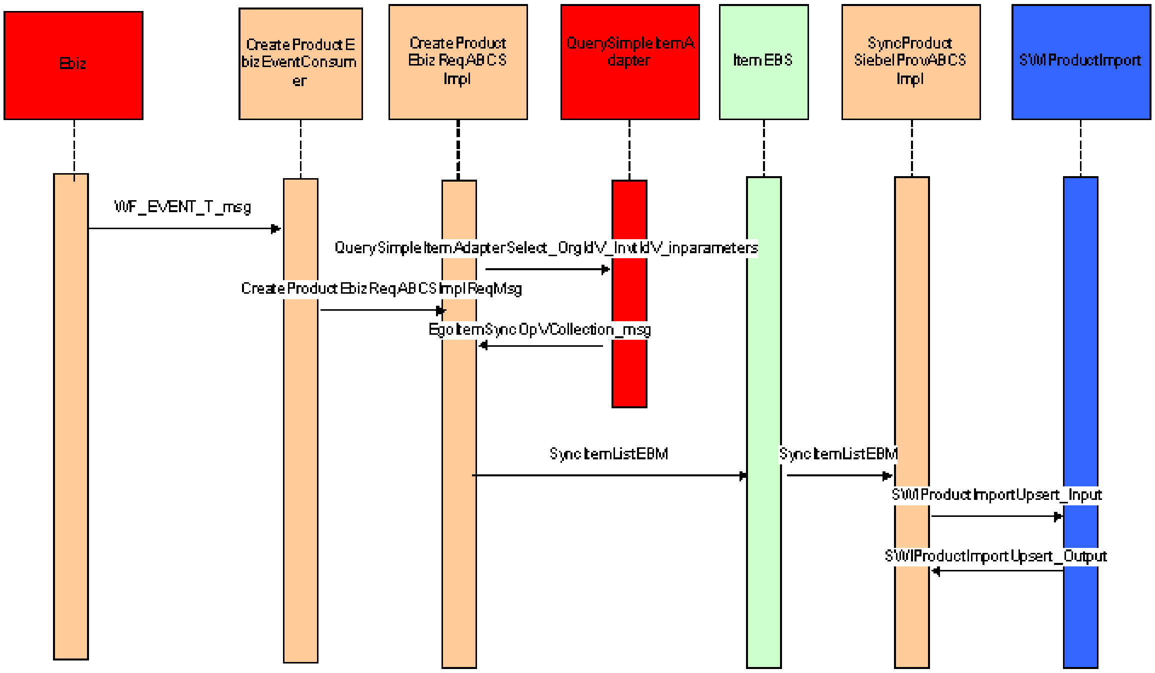
Synchronize Items integration flow

Create Items Flow

This integration flow uses the following interfaces:

- CreateProductEbizReqABCSImpl
- ItemEBS
- SyncProductSiebelProvABCSImpl
- ItemResponseEBS

This sequence diagram illustrates the Create Item integration flow:


Ebiz
CreateProductEbizEventConsumer
CreateProductEbizReqABCSImpl
QuerySimpleItemAdapter
ItemEBS
SyncProductSiebelProvABCSImpl
SWIProductImport
WF_EVENT_T_msg
QuerySimpleItemAdapterSelect_OrgIdV_InvtIdV_inparameters
CreateProductEbizReqABCSImplReqMsg
EgoItemSyncOpVCollection_msg
SyncItemListEBM
SyncItemListEBM
SWIProductImportUpsert_Input
SWIProductImportUpsert_Output

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.

   a 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.

   b As part of the transformation process, the CreateProductEbizReqABCSImpl transforms the EgoItemSyncOpVCollection_msg message into the SyncItemListEBM.

   Invoke the ItemEBS with the operation SyncItemList. The ItemEBS is a routing ESB service with several operations on the Item enterprise business message (EBM). In the process, the cross-reference is created by concatenating 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 Siebel product web service SWIProductImport to synchronize the product.

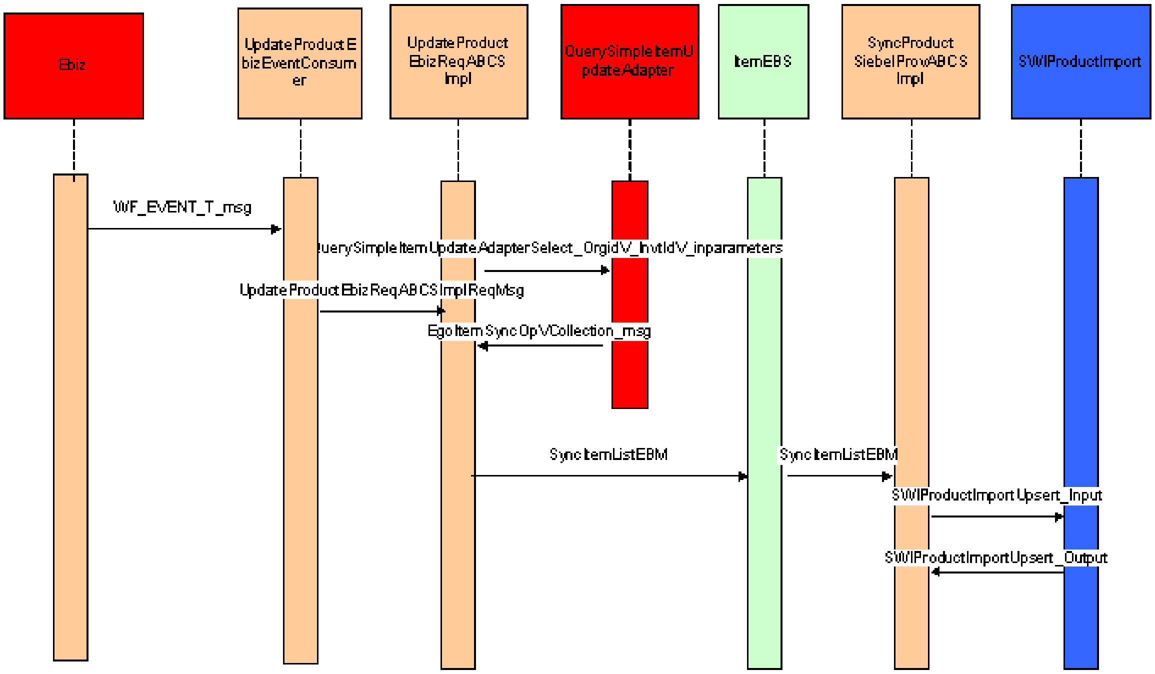
The Siebel web service completes the request and returns a response message. In the process, the cross-reference is linked with the Siebel IDs of the product.

Update Items Integration Flow

This integration flow uses the following interfaces:

- UpdateProductEbizReqABCSImpl
- ItemEBS
- SyncProductSiebelProvABCSImpl

This sequence diagram illustrates the Update Items integration flow:


Ebiz
UpdateProductEbizEventConsumer
UpdateProductEbizReqABCSImpl
QuerySimpleItemUpdateAdapter
ItemEBS
SyncProductSiebelProvABCSImpl
SWIProductImport
WF_EVENT_T_msg
uerySimpleItemUpdateAdapterSelect_OrgidV_InvtIdV_inparameters
UpdateProductEbizReqABCSImplReqMsg
EgoItemSyncOpVCollection_msg
SyncItemListEBM
SyncItemListEBM
SWIProductImportUpsert_Input
SWIProductImportUpsert_Output

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 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.

   a 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.

b. As part of the transformation process, the UpdateProductEbizReqABCSImpl transforms the EgoItemSyncOpVCollection_msg message into the SyncItemListEBM.

c. Invoke the ItemEBS with the operation SyncItemList.

   The ItemEBS is a routing ESB service with several operations on the Item EBM.

d. In the process, the cross-reference is created by concatenating 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 with the Siebel IDs of the Product.

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 needs to be synchronized. This flow is designed to be used for incremental loads; it is not intended to be used for high-volume initial loads.

For more information about initial data loads, see Chapter 2: Describing the Initial Data Loads

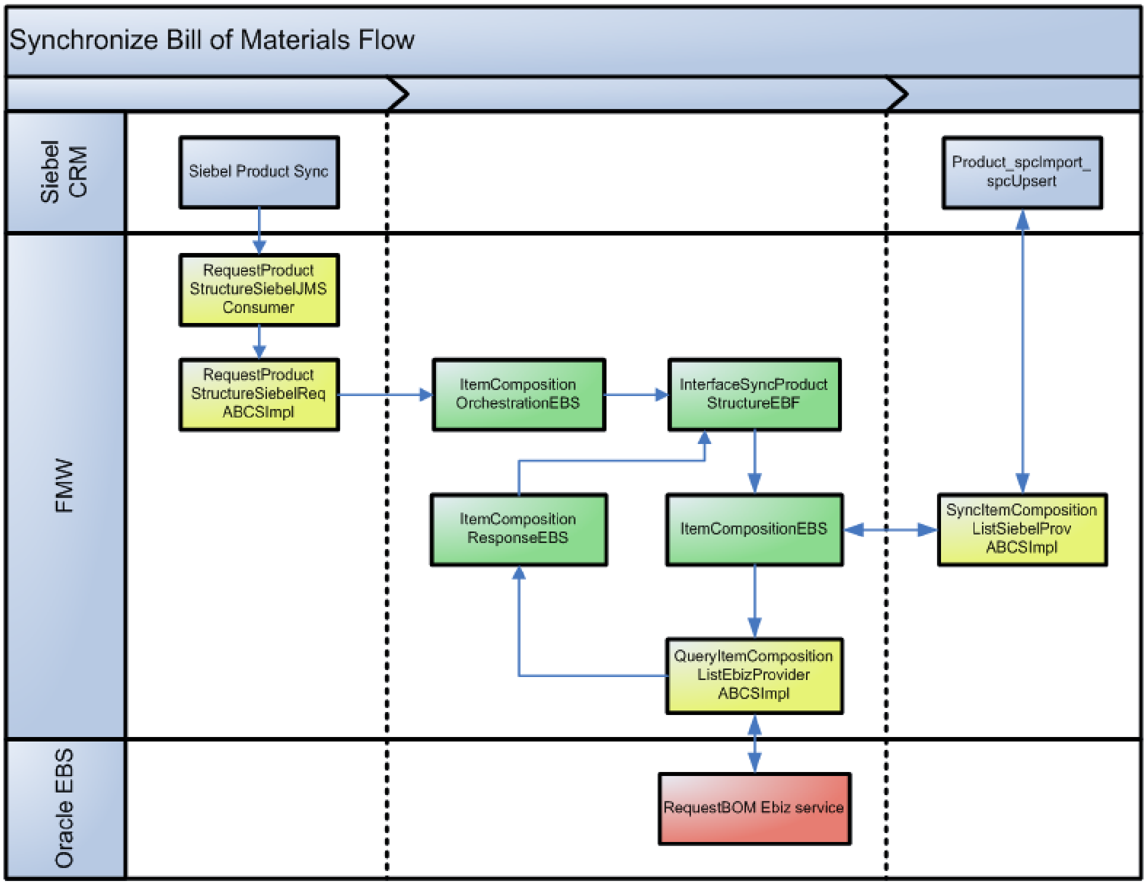
The Synchronize BOM integration flow brings over all non-orderable items and item relationships based on the requested BOM structure.

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:

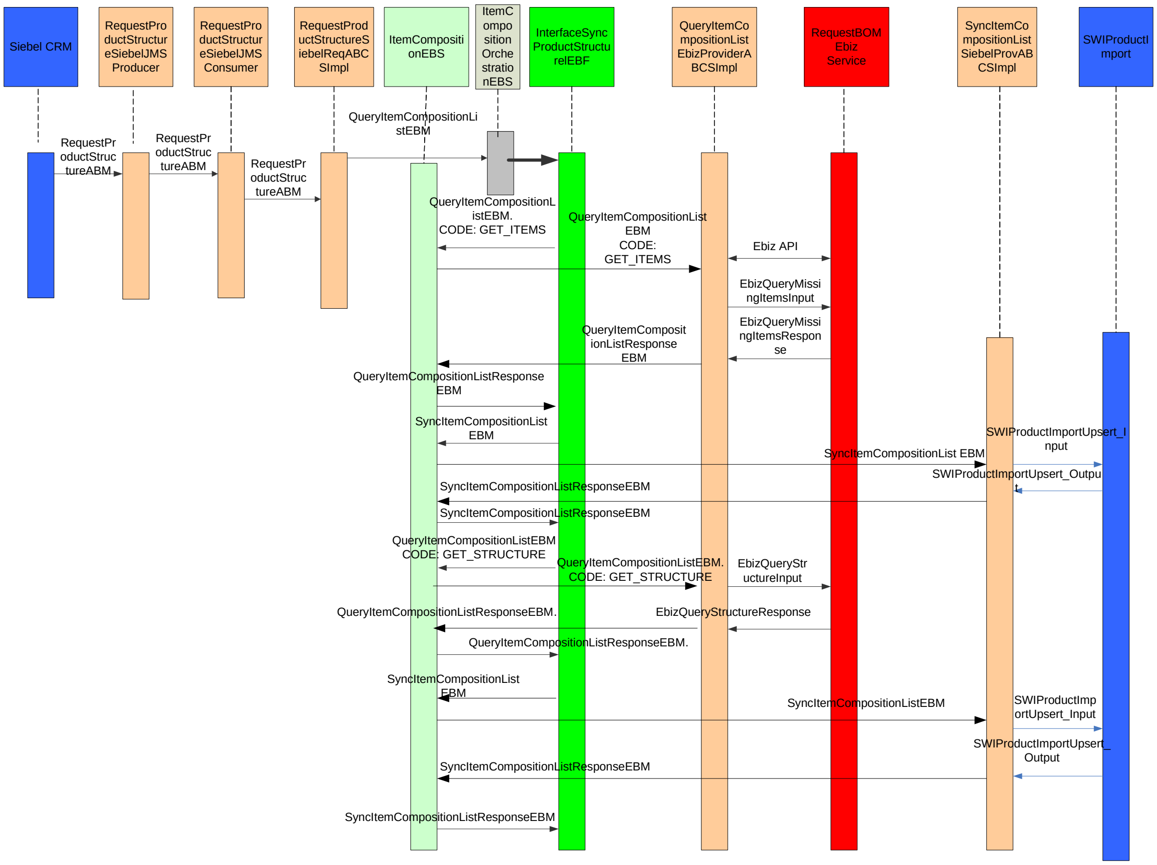
- RequestProductStructureSiebelReqABCSImpl
- ItemCompositionOrchestrationEBS
- ItemCompositionEBS
- ItemCompositionResponseEBS
- InterfaceSyncProductStructurelEBF
- QueryItemCompositionListEbizProviderABCSImpl
- SyncItemCompositionListSiebelProvABCSImpl

This diagram illustrates the flow for synchronizing BOMs from Oracle EBS to Siebel CRM:


Synchronize Bill of Materials Flow
Siebel CRM
FMW
Oracle EBS
Siebel Product Sync
RequestProduct StructureSiebelJMS Consumer
RequestProduct StructureSiebelReq ABCSImpl
ItemComposition OrchestrationEBS
InterfaceSyncProduct StructureEBF
ItemComposition ResponseEBS
ItemCompositionEBS
QueryItemComposition ListEbizProvider ABCSImpl
RequestBOM Ebiz service
Product_spcImport_ spcUpsert
SyncItemComposition ListSiebelProv ABCSImpl

Synchronize BOM integration flow

This sequence diagram illustrates the Synchronize BOM integration flow:


Siebel CRM
RequestProductStructureSiebelJMSProducer
RequestProductStructureSiebelJMSConsumer
RequestProductStructureSiebelReqABCSImpl
ItemCompositionEBS
ItemCompositionOrchestrationEBS
InterfaceSyncProductStructureIEBF
QueryItemCompositionListEbizProviderABCSImpl
RequestBOM Ebiz Service
SyncItemCompositionListSiebelProvABCSImpl
SWIProductImport
RequestProductStructureABM
RequestProductStructureABM
RequestProductStructureABM
QueryItemCompositionListEBM
QueryItemCompositionListEBM. CODE: GET_ITEMS
QueryItemCompositionList EBM CODE: GET_ITEMS
Ebiz API
EbizQueryMissingItemsInput
EbizQueryMissingItemsResponse
QueryItemCompositionListResponse EBM
QueryItemCompositionListResponse EBM
SyncItemCompositionList EBM
SWIProductImportUpsert_Input
SyncItemCompositionList EBM
SWIProductImportUpsert_Output
SyncItemCompositionListResponseEBM
SyncItemCompositionListResponseEBM
QueryItemCompositionListEBM CODE: GET_STRUCTURE
QueryItemCompositionListEBM. CODE: GET_STRUCTURE
EbizQueryStructureInput
QueryItemCompositionListResponseEBM.
EbizQueryStructureResponse
QueryItemCompositionListResponseEBM.
SyncItemCompositionList EBM
SyncItemCompositionListEBM
SWIProductImportUpsert_Input
SWIProductImportUpsert_Output
SyncItemCompositionListResponseEBM
SyncItemCompositionListResponseEBM

Synchronize BOM flow sequence diagram

When you initiate the process, the following events occur:

1. The SyncProductStructureSiebelJMSProvider enqueues the Product ID, Inventory Location ID, Business Unit ID to the 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 dequeue 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 ESB service that routes the request to the InterfaceSyncProductStructureIEBF enterprise business flow.

4. The InterfaceSyncProductStructureIEBF 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 and then calls the Oracle 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 non-orderable items from the back office.

   Next, it transforms the Oracle response message EbizQueryMissingItemsResponse into 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 concatenating 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 is linked with 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 InterfaceSyncProductStructureIEBF.

10. The InterfaceSyncProductStructureIEBF 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 InterfaceSyncProductStructureIEBF.

Data Requirements

This process integration has no data requirements.

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 ABM. The ABM definition includes a list of product IDs.

For more information about Siebel web services, see Siebel CRM Integration Pack for Oracle Order Management Addendum

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 EBS web services, see the following Oracle EBS references: E-Business Suite Electronic Technical Reference Manual (eTRM) located on My Oracle Support under the E-Business Suite Information Center, Oracle Integration Repository located at http://irep.oracle.com, and Oracle Applications Release 11.5.10+ Online Documentation Library, located on the Oracle Technology Network (http://www.oracle.com/technology/documentation/applications.html).

Core AIA Components

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

- ItemCompositionEBO
- ItemCompositionListEBM
- ItemEBO
- ItemListEBM
- ItemEBSV2
- ItemCompositionOrchestrationEBS
- InterfaceSyncProductStructureEBF
- ItemCompositionEBS
- ItemCompositionResponseEBS

The core EBO and EBM XSD files can be located by EBO within this parent folder: http://[HOST:PORT]/AIAComponents/EnterpriseObjectLibrary/Core/EBO/.

The core EBS WSDL files can be located by EBO within this parent folder: http://[HOST:PORT]/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/.

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

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

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 Application Integration Architecture - Foundation Pack: Integration Developer’s Guide*, “Extensibility for Oracle AIA Artifacts”

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

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

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

ItemEBSV2

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

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 up 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 *Oracle Application Integration Architecture - Foundation Pack: Integration Developer's Guide*, "Designing and Developing EBSs" and *Oracle Application Integration Architecture - Foundation Pack: Concepts and Technologies Guide*, "Understanding EBSs"

ItemCompositionResponseEBSV2

The ItemCompositionResponseEBSV2 is a lightweight ESB 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: Siebel CRM – EBS 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.

ItemResponseEBSV2

The ItemResponseEBSV2 is a lightweight ESB 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.

CreateProductEbizReqABCSImpl

The CreateProductEbizReqABCSImpl 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 SyncItem EBS operation. By default, an Orderable Product associated with the OE:Item Validation Org will be 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.

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.

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.

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 released product in Siebel CRM is controlled using the configuration property:

- 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.

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.

ItemCompositionOrchestrationEBS

The ItemCompositionOrchestrationEBS is a lightweight ESB routing service that routes the request from the RequestProductStructureSiebelReqABCSImpl service to 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.

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 it to Siebel CRM. This service is invoked from the ItemCompositionOrchestrationEBS service with request message QueryItemCompositionListEBM.

There is a single operation to 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.)

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 executing Oracle API BOM_STRUCT_SYNC_PUB.EXPLODE_STRUCTURE through Oracle Apps adapter to explode the BOM.

By default, the QueryItemCompositionListEbizProvABCSImpl publishes all non-orderable child components of the BOM and BOM Structure. This function is controlled using the configuration property PUBLISH_ALL_PRODUCT.

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

ItemCompositionEBS

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

ItemCompositionEBS provides these operations:

- QueryItemCompositionList
- SyncItemCompositionList

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

SyncItemCompositionList

When a Siebel Product Administrator requests one or more product structures to be available in 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.

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 will be synchronized to Siebel as Bundle Product and will be released as part of synchronization process.

Release Product in Siebel is controlled using these configuration properties:

- 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.

RequestProductStructureSiebelJMSProducer

The RequestProductStructureSiebelJMSProducer is implemented as a BPEL process because it involves JMS Header manipulations that cannot be done in ESB.

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

RequestProductStructureSiebelJMSConsumer

The RequestProductStructureSiebelJMSConsumer is implemented as an ESB 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: RequestProductStructureJMSConsumer

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”: CreateItemEbizEventConsumer

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

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”: UpdateItemEbizEventConsumer

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

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”: BulkloadItemEbizEventConsumer

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

Chapter 5: Understanding Process Integration for Price Lists

This chapter discusses:

- Process integration for Price Lists
- Loading initial price lists
- Loading incremental price lists

Process Integration for Price Lists

The process integration for price lists is different from the other integrations in the Order to Cash: Siebel CRM – EBS Process Integration Pack (Order to Cash: Siebel CRM – EBS PIP). 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 EBS to Siebel CRM.

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

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

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

- **Incremental load** 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: Describing the Initial Data Loads

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 could 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 or not the price lists or the header and line information already 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 needed.

If Pricing Security is turned on and if a price list is not global, you need to 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. A nonglobal price list can be used 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.

Prerequisite

The prerequisite for the process integration for Price Lists is product synchronization.

For more information about product synchronization, see Chapter 4: Describing the Process Integration for Product Management

Solution Assumptions and Constraints

The integration design assumes that the following statements are true:

1. Only a bulk load process is supported.

   a ODI pulls all of the order-related active prices lists from Oracle EBS into Siebel CRM.

   b To synchronize incremental loads, run the same process but specify the price list names that will 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 already 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, end date) and price list line information (such as product, list price, start date, end date) are synchronized.

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

Loading Initial Price Lists

For more information about initial data loads, see Chapter 2: Describing the Initial Data Loads

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.

Loading Price List Data from Oracle EBS to Siebel EIM Overview

Pricing Administrators are responsible to:

1. Specify the list of price list names in the XML file at $AIA_HOME/staging/O2C2Home/SetupFiles/PriceListSource.xml for the incremental load.
2. Populate 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. Load the specified price lists, new or updated, into the Siebel interface tables (EIM_PRI_LST) by running an ODI package.
4. Run the Siebel EIM program to populate the Siebel base tables.
5. Link cross-references with the Siebel ID by running an ODI package.

For more information about initial data loads, see Chapter 2: Describing the Initial Data Loads

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:

   a ODI package: LoadEbizIncPriceListDataToSiebelPkg

   b ODI Scenario: LOADEBIZINCPRICELISTDATATOSIEBEL

   c ODI Interface: Ebiz_To_Xref_PriceList_INCINTERFACE d Source: CRMINTEG_PRICELIST_V

e Target: cross-reference table

f Join condition: Select * from QP_CRMINTEG_PRICELIST_V

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:

a ODI package: LoadEbizIncPriceListDataToSiebelPkg

b ODI Scenario: LOADEBIZINCPRICELISTDATATOSIEBEL

c ODI Interface: Ebiz_To_Sebl_PriceList_INCINTERFACE

d Source: QP_CRMINTEG_PRICELIST_V

e Join condition: Select * from QP_CRMINTEG_PRICELIST_V where name in (price list names mentioned in the xml)

f Target: EIM_PRI_LST

Siebel EIM Configuration File

The configuration file follows:

```
[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

Loading the Cross-Reference Table

After Siebel EIM tables are populated, you run the Siebel EIM program 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: Describing the Initial Data Loads

Source Schema Description

- S_PRI LST

Target Schema Description

- X-REF table

ODI Mapping Design

- S_PRICE LIST table has ROW_ID and INTEGRATION_ID that need to 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

Column in Target	Column in Source (Table)
SEBL_01	ROW_ID (S_PRI LST)
ORCL_01	Should already be populated during the Oracle view to EIM map.
COMMON	INTEGRATION_ID (S_PRI_LST)

Chapter 6: Understanding Process Integration for Quotes

This chapter discusses:

- Process integration for Quotes
- Quotes integration flow
- Data requirements
- Siebel Customer Relationship Management (CRM) interfaces
- Oracle E-Business Suite (Oracle EBS) interfaces
- Core Oracle Application Integration Architecture (AIA) components
- Integration services

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. You use Siebel CRM to capture quotes. Since Siebel CRM also handles pricing, you can create and display quotes 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 needed. 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, it is customized using Oracle Configurator. Siebel CRM and Oracle Configurator are seamlessly integrated.

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

Prerequisites

The prerequisites for the Quotes integration are:

- Product synchronization
- Pricing synchronization

For more information, see Chapter 4: Describing the Process Integration for Product Management and Chapter 5: Describing the Process Integration for Price Lists

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. No further updates to it in Siebel CRM are permitted.
2. Quotes have to be in approved status before they can be submitted to the back office.
3. B2C scenarios are not supported by this integration.
4. No support for ATP Check, Credit Check, and Shipping Charges for Quotes is delivered with the initial product.

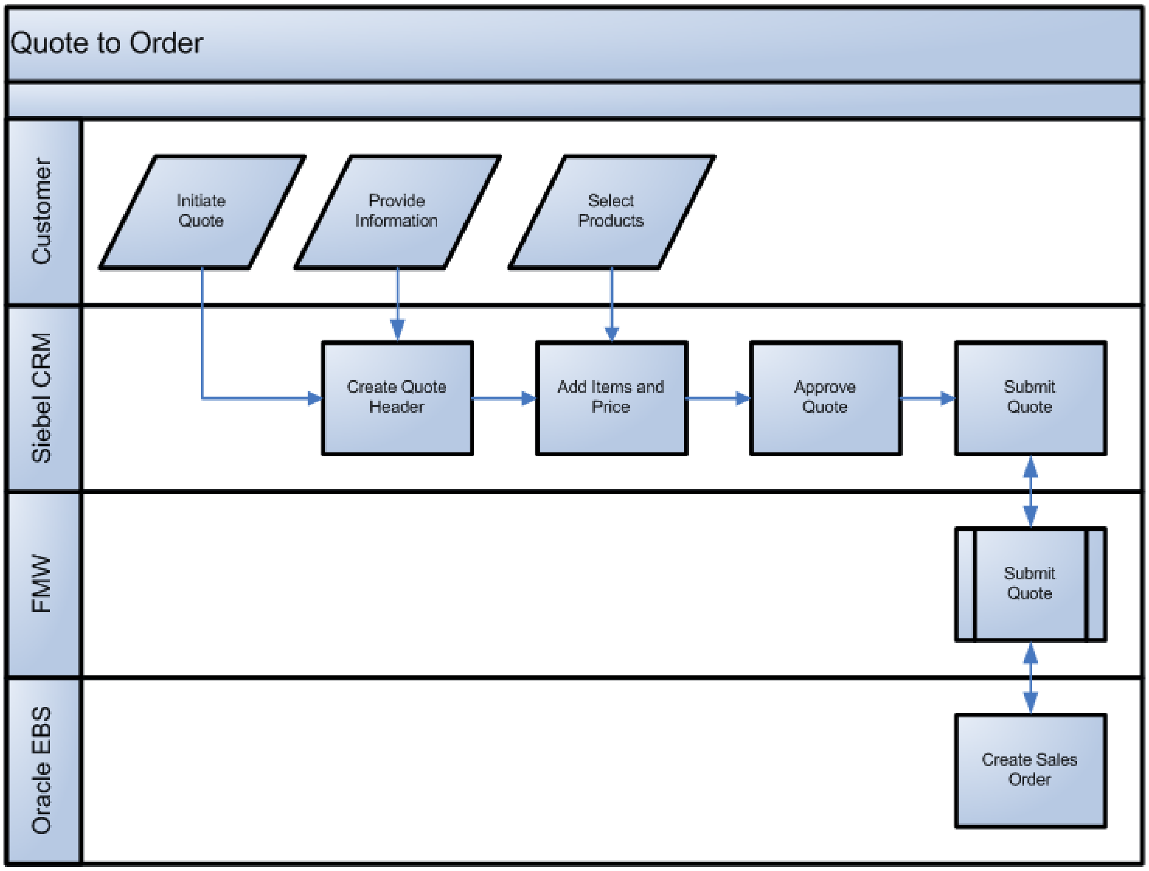
Quotes Integration Flow

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

You use Siebel CRM to determine pricing. Siebel CRM also enables you to create quotes immediately without having to get pricing information from a back office system.

You then give the quote details to the customer. After the customer approves the quote, you submit it to the back office for order fulfillment.

This diagram illustrates the flow for the Quotes process integration:



Quotes process integration

Data Requirements

Data requirements indicate the mandatory data that must be provided to make the flow successful:

- The Quotes integration flow must have a minimum of one quote line item, and the quote line item must have a product that was already synchronized from the back office.
- The Quote must have valid customer information.

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 ABM

For more information about Siebel web services, see the Siebel CRM Integration Pack for Oracle Order Management Addendum

Oracle EBS Interfaces

The Quotes integration flow uses the following Oracle EBS service:

Inbound to EBS Web Service

- OE_INBOUND_INT.PROCESS_ORDER (Process Sales Order Service)

For more information about EBS web services, see the following Oracle EBS references:
E-Business Suite Electronic Technical Reference Manual (eTRM) located on My Oracle Support under the E-Business Suite Information Center,
Oracle Integration Repository located at http://irep.oracle.com,
Oracle Applications Release 11.5.10+ Online Documentation Library, located on the Oracle Technology Network (http://www.oracle.com/technology/documentation/applications.html)

Core AIA Components

The Quotes integration uses the following delivered horizontal components:

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

The core enterprise business object (EBO) and enterprise business message (EBM) XSD files can be located by EBO within this parent folder:
http://[HOST:PORT]/AIAComponents/EnterpriseObjectLibrary/Core/EBO/.

The core EBS WSDL files can be located by EBO within this parent folder: http://[HOST:PORT]/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/.

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

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

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

For more information, see *Oracle Application Integration Architecture - Foundation Pack: Integration Developer’s Guide*, “Extensibility for Oracle AIA Artifacts”

Integration Services

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

- ProcessQuoteSiebelJMSProducer
- ProcessQuoteSiebelJMSConsumer
- ProcessQuoteSiebelReqABCSImpl

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

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

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

ProcessQuoteSiebelJMSProducer

The ProcessQuoteSiebelJMSProducer service is responsible for providing guaranteed delivery of the ProcessQuote Siebel ABM to the JMS queue. This service is invoked synchronously from the Siebel application workflow. The response will indicate whether the message was successfully enqueued. This service is invoked as part of the Create Quote integration flow.

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

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

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, when the version of the Siebel CRM application is 8.0.0.7 or lower.

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

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

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.

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

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

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, 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 is implemented as an asynchronous request-only BPEL process.

Chapter 7: Understanding the Available to Promise Check Integration Flow

This chapter discusses:

- ATP Check requests
- Data requirements
- Siebel Customer Relationship Management (CRM) interfaces
- Oracle E-Business Suite (Oracle EBS) interfaces
- Core Oracle Application Integration Architecture (AIA) components
- Integration services

ATP Check Requests

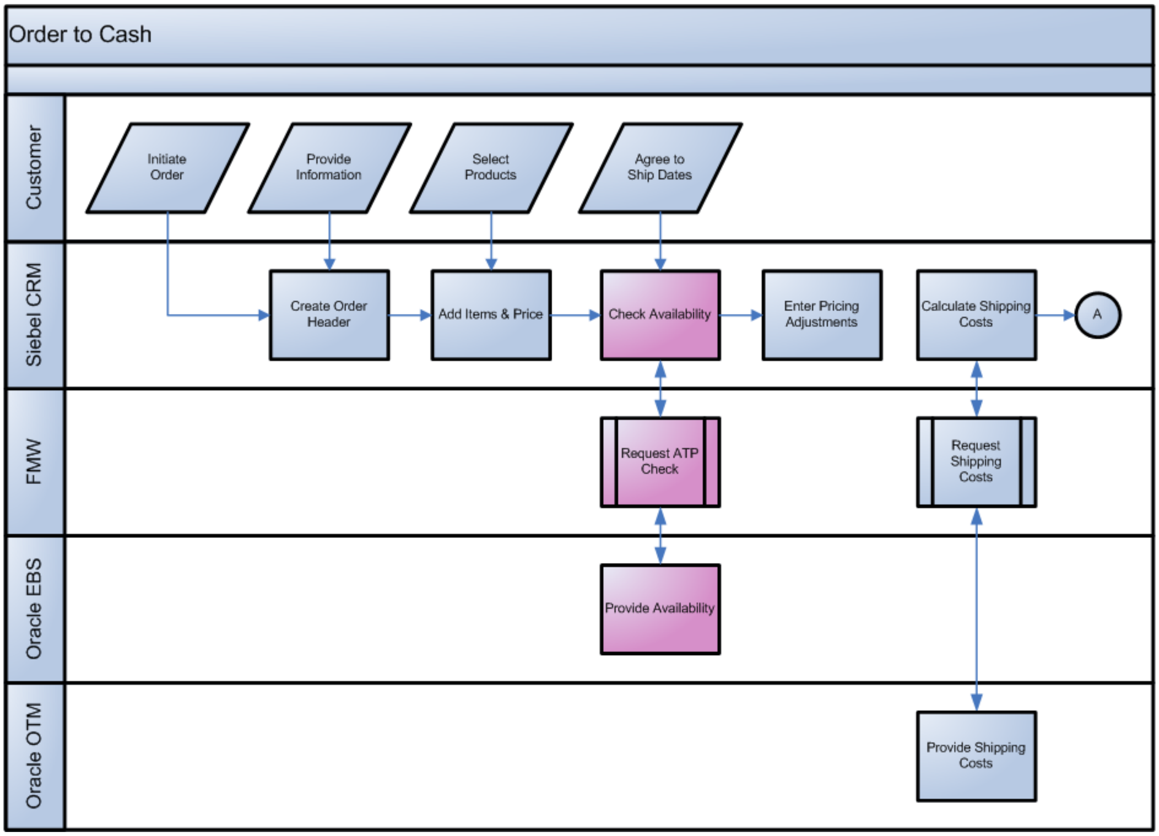
You can call the ATP Check integration flow on request before you call the Create Order integration flow. The ATP Check integration flow initiates in Siebel CRM to obtain product availability quantities and dates from Oracle EBS. This data is returned to Siebel CRM so that a Customer Service Representative (CSR) can inform the customer and continue with the order creation process. Calling the ATP Check before an order is submitted increases the chances of successfully fulfilling the order from the back office. Correct setting of customer expectations 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.

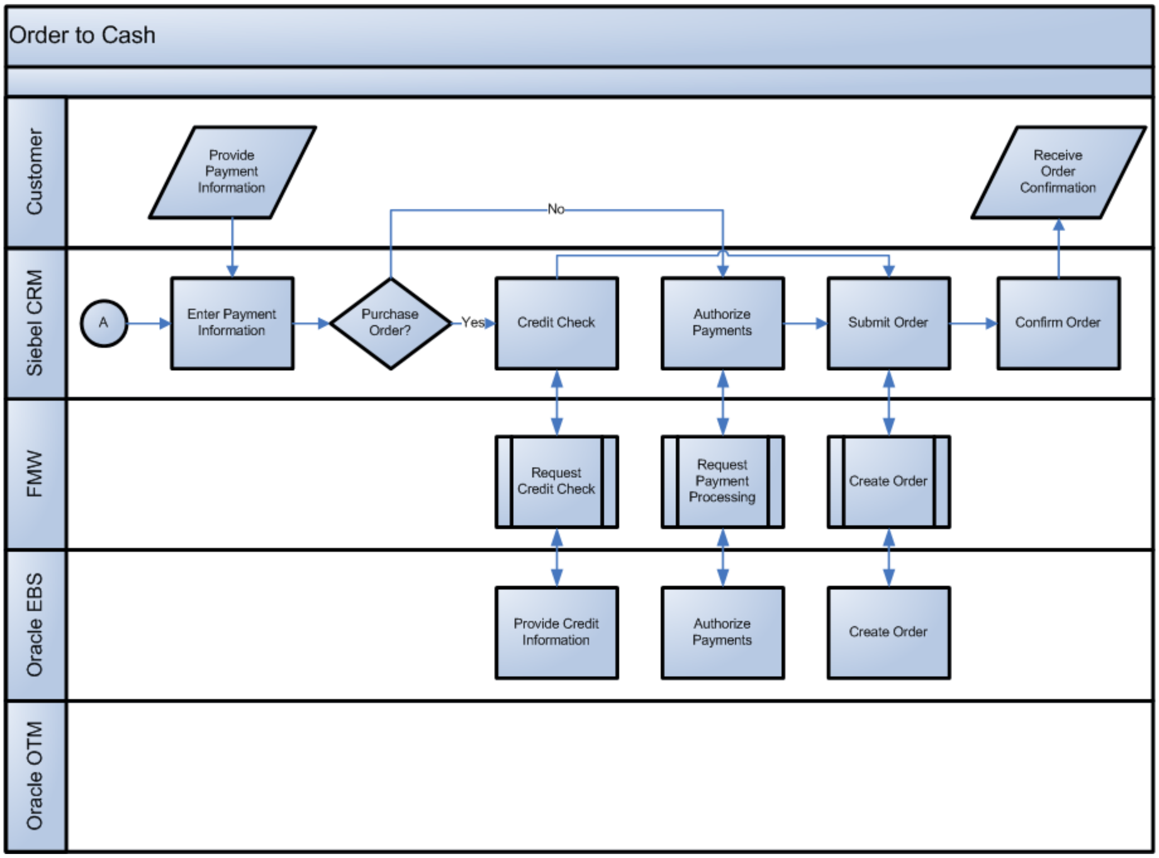
Note: Inventory locations must be manually maintained in both systems, and they must have the same name and address. Inventory Location IDs are manually cross-referenced.

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

The following diagrams illustrate the where ATP Check integration flow fits in the Order to Cash: Siebel CRM – EBS integration flow:



ATP Check integration flow (1 of 2)



ATP Check integration flow (2 of 2)

Prerequisites

The prerequisite for the ATP Check integration flow is product synchronization.

For more information about the product synchronization flow, see Chapter 4: Describing the Process Integration for Product Management

Solution Assumptions and Constraints

These are the assumptions for the ATP Check integration flow:

1. ATP Check is only supported for simple products (that is, Items).

   ATP Check cannot be performed against a complex product (such as BOM) structure.

2. Available quantity at the line level appears only when the status is Available.

   For any other ATP status, the value at the line level is not populated. If the status is Out of Stock, ATP not applicable, or Plan not found, then the Available Quantity shown at the line level is blank and the actual available status appears at the Promise line level.

3. The default requested date that appears in the Siebel CRM fulfillment window is the system date plus one day.

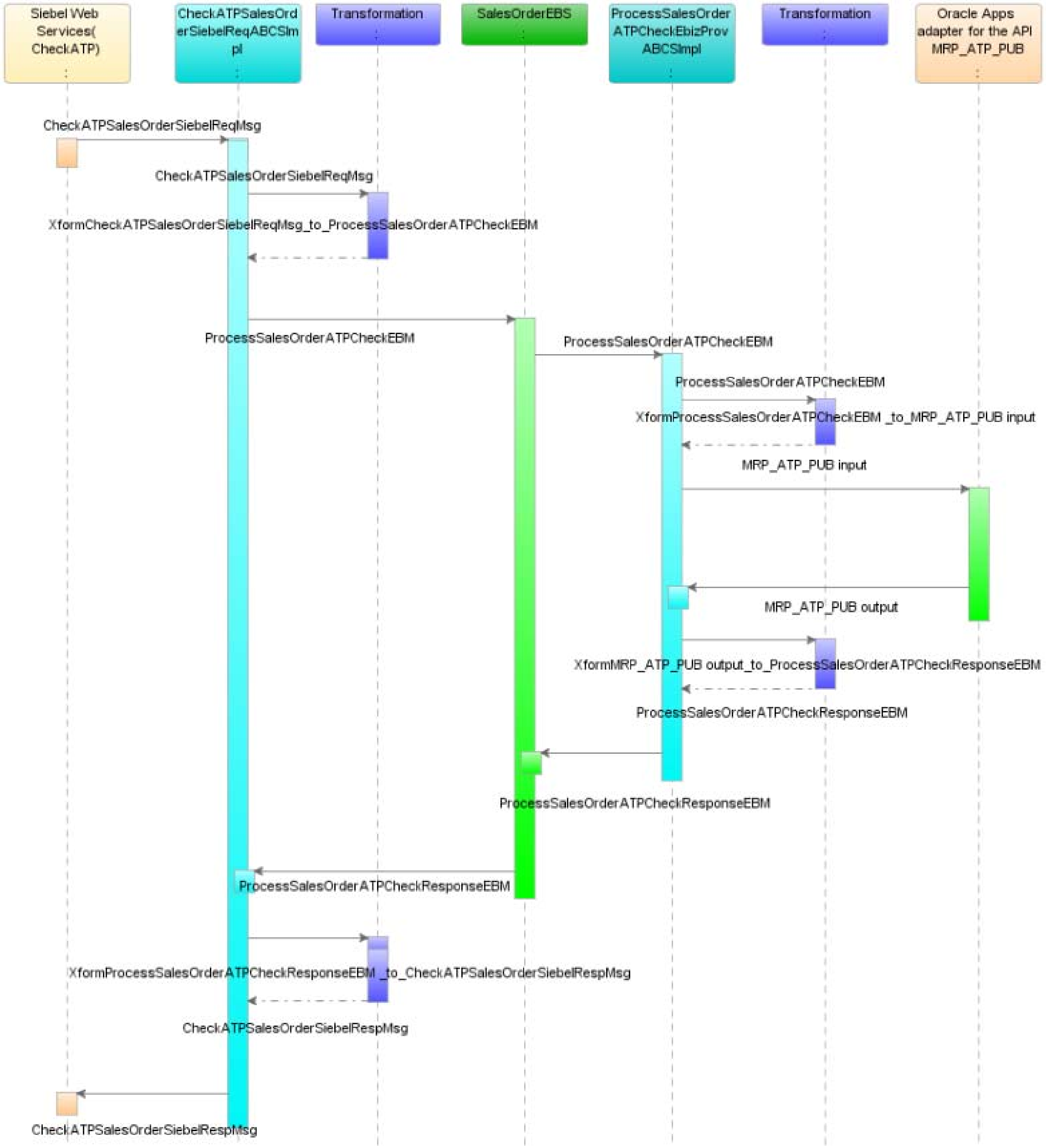
   This value is used for the ATP check, and the resultant availability reflects this date.

ATP Check Integration Flow

This integration flow uses the following interfaces:

- CheckATPSalesOrderSiebelReqABCSImpl
- SalesOrderEBS
- ProcessSalesOrderATPCheckEbizProvABCSImpl

This sequence diagram illustrates the ATP Check integration flow:


Siebel Web Services(CheckATP)
CheckATPSalesOrderSiebelReqABCSImpl
Transformation
SalesOrderEBS
ProcessSalesOrderATPCheckEbizProvABCSImpl
Transformation
Oracle Apps adapter for the API MRP_ATP_PUB
CheckATPSalesOrderSiebelReqMsg
CheckATPSalesOrderSiebelReqMsg
XformCheckATPSalesOrderSiebelReqMsg_to_ProcessSalesOrderATPCheckEBM
ProcessSalesOrderATPCheckEBM
ProcessSalesOrderATPCheckEBM
ProcessSalesOrderATPCheckEBM
XformProcessSalesOrderATPCheckEBM _to_MRP_ATP_PUB input
MRP_ATP_PUB input
MRP_ATP_PUB output
XformMRP_ATP_PUB output_to_ProcessSalesOrderATPCheckResponseEBM
ProcessSalesOrderATPCheckResponseEBM
ProcessSalesOrderATPCheckResponseEBM
ProcessSalesOrderATPCheckResponseEBM
XformProcessSalesOrderATPCheckResponseEBM _to_CheckATPSalesOrderSiebelRespMsg
CheckATPSalesOrderSiebelRespMsg
CheckATPSalesOrderSiebelRespMsg

ATP Check flow sequence diagram

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

   a In the Order window, click the Fulfillment tab.

   b A line item appears on the line item applet.

   c Navigate to the line items applet.

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

d 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 with the message ProcessSalesOrderATPCheckResponseEBM and the operation ProcessSalesOrderATPCheck routes it to the invoking CheckATPSalesOrderSiebelReqABCSImpl.
8. The CheckATPSalesOrderSiebelReqABCSImpl transforms the ProcessSalesOrderATPCheckResponseEBM into the CheckATPSalesOrderRespMsg and the response is sent back to the invoking Siebel web service.
9. If successful, the ATP amounts appear in Siebel CRM.

Data Requirements

Data requirements indicate the mandatory data that you must provide to make the flow successful:

- Order has at least one line item.
- Order Qty is > 0.
- A product that has been synchronized from EBS is specified.
- Requested delivery date is specified.

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 the Siebel CRM Integration Pack for Oracle Order Management Addendum

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 EBS web services, see the following Oracle EBS references:
E-Business Suite Electronic Technical Reference Manual (eTRM) located on My Oracle Support under the E-Business Suite Information Center,
Oracle Integration Repository located at http://irep.oracle.com, Oracle Applications Release 11.5.10+ Online Documentation Library, located on the Oracle Technology Network (http://www.oracle.com/technology/documentation/applications.html)

Core AIA Components

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

- SalesOrderEBO
- ProcessSalesOrderATPCheckEBM
- ProcessSalesOrderATPCheckResponseEBM
- SalesOrderEBS

The core EBO and enterprise business message (EBM) XSD files can be located by EBO within this parent folder: http://[HOST:PORT]/AIAComponents/EnterpriseObjectLibrary/Core/EBO/.

The core EBS WSDL files can be located by EBO within this parent folder: http://[HOST:PORT]/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/.

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

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

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

For more information, see *Oracle Application Integration Architecture - Foundation Pack: Integration Developer’s Guide*, “Extensibility for Oracle AIA Artifacts”

Integration Services

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

- SalesOrderEBS
- CheckATPSalesOrderSiebelReqABCSImpl
- ProcessSalesOrderATPCheckEbizProvABCSImpl

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

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

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 Application Integration Architecture - Foundation Pack: Integration Developer’s Guide*, “Designing and Developing EBSs” and *Oracle Application Integration Architecture - Foundation Pack: Concepts and Technologies Guide*, “Understanding EBSs”

CheckATPSalesOrderSiebelReqABCSImpl

The CheckATPSalesOrderSiebelReqABCSImpl is the Application Business Connector (ABC) Service 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

ProcessSalesOrderATPCheckEbizProvABCSImpl

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

The ProcessSalesOrderATPCheckEbizProvABCSImpl transforms the ProcessSalesOrderATPCheckEBM message to 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.

Chapter 8: Understanding the Shipping Charges Integration Flow

This chapter discusses:

- Shipping Charges integration flow
- Data requirements
- Siebel Customer Relationship Management (CRM) interfaces
- Oracle Transportation Management (OTM) interfaces
- Core Oracle Application Integration Architecture (AIA) components
- Integration services

Shipping Charges Integration Flow

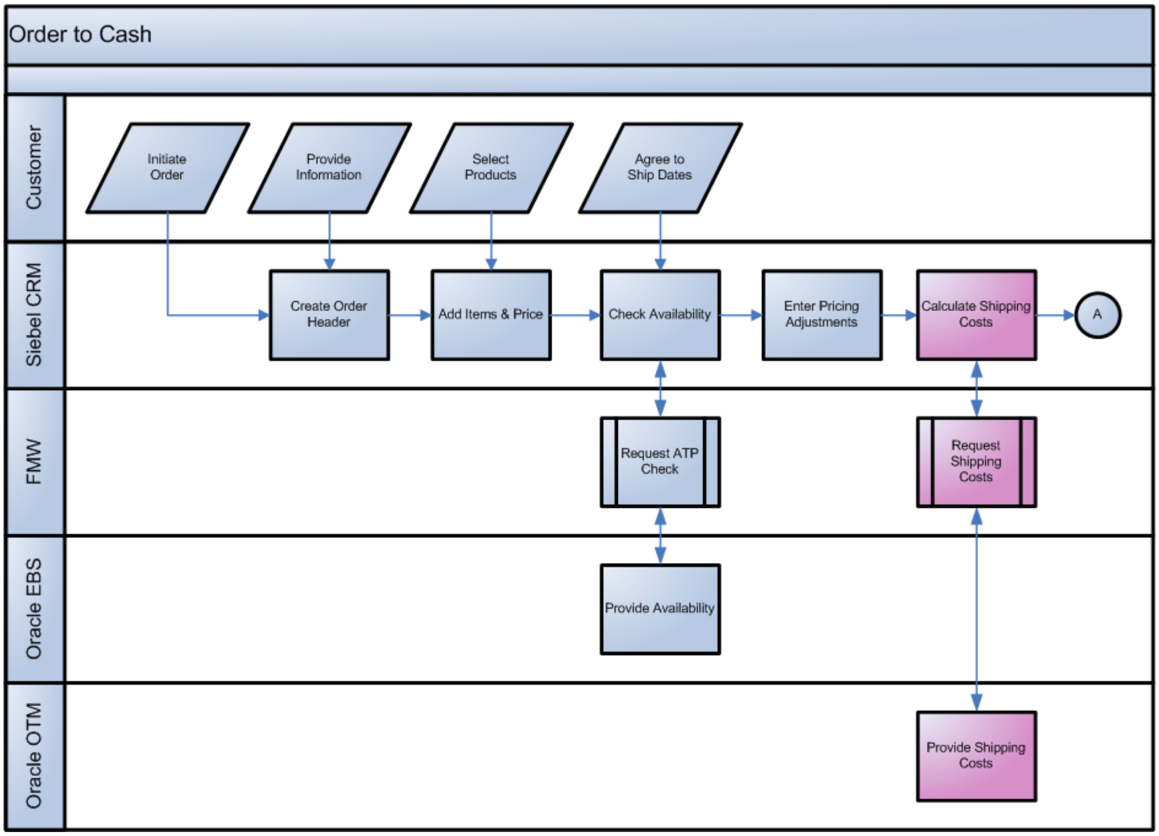
The Shipping Charges integration flow can be called on request, prior to calling the Create Order integration flow. The Shipping Charges integration flow is initiated in Siebel CRM and obtains the estimated shipping cost from Oracle EBS. The source address, destination address, and weight information is passed to Oracle Transportation Management (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.

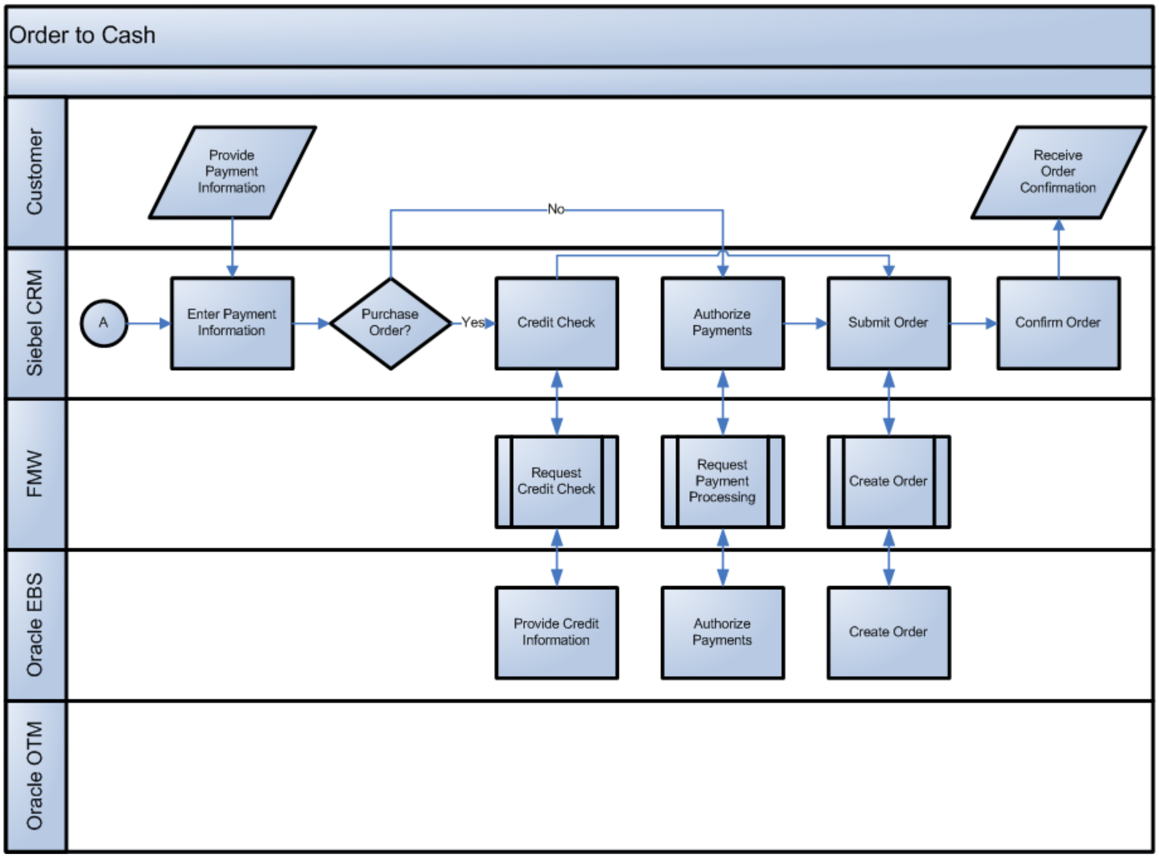
The address, weight, and inventory source information are transferred to OTM to calculate and return the shipping charge amount. The Actual Freight charge calculations occur only at the time of shipping in Oracle EBS.

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

These diagrams illustrate where the Shipping Charges integration flow fits in the Order to Cash: Siebel CRM – EBS integration flow and:



Shipping Charges integration flow (1 of 2)



Shipping Charges integration flow (2 of 2)

Prerequisite

For the Shipping Charges integration flow to work successfully, you must set a property for OTM 5.5 CU5.

For more information about setting this property, see Setting a Property in OTM

The prerequisite for the Shipping Charges integration flow is product synchronization.

For more information about the product synchronization flow, see Chapter 4: Describing the Process Integration for Product Management

Solution Assumptions and Constraints

These are the assumptions for the Shipping Charges integration flow:

1. Shipping charges are not supported for complex products (such as BOM).
2. Shipping charges are not supported for Quotes.

3. The only unit of measure (UOM) that is available is physical weight. Many aspects can determine the costs of the shipping, for example, volume, the nature of the entity, and so on, which OTM supports; however, Siebel is unable to support this feature.

   The UOM is picked up as a default value from the AIAConfigurationProperties.xml file.

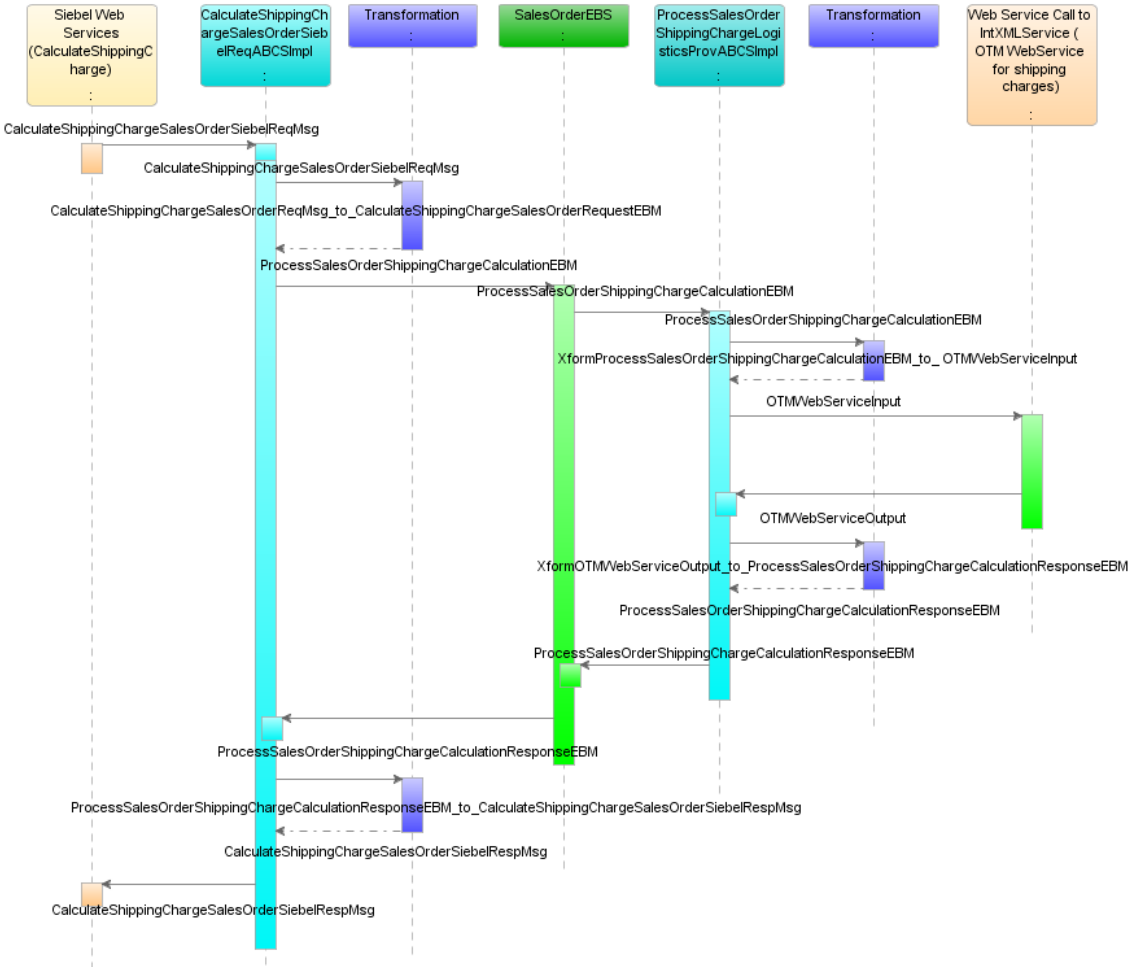
4. The actual freight that is determined when shipped in Oracle EBS is not returned to Siebel CRM.

Shipping Charges Integration Flow

This integration flow uses the following interfaces:

- SalesOrderEBS
- CalculateShippingChargeSalesOrderSiebelReqABCSImpl
- ProcessSalesOrderShippingChargeLogisticsProvABCSImpl

This sequence diagram illustrates the Shipping Charges integration flow:


Siebel Web Services (CalculateShippingCharge)
CalculateShippingChargeSalesOrderSiebelReqABCSImpl
Transformation
SalesOrderEBS
ProcessSalesOrderShippingChargeLogisticsProvABCSImpl
Transformation
Web Service Call to IntXMLService (OTM WebService for shipping charges)
CalculateShippingChargeSalesOrderSiebelReqMsg
CalculateShippingChargeSalesOrderSiebelReqMsg
CalculateShippingChargeSalesOrderReqMsg_to_CalculateShippingChargeSalesOrderRequestEBM
ProcessSalesOrderShippingChargeCalculationEBM
ProcessSalesOrderShippingChargeCalculationEBM
ProcessSalesOrderShippingChargeCalculationEBM
XformProcessSalesOrderShippingChargeCalculationEBM_to_ OTMWebServiceInput
OTMWebServiceInput
OTMWebServiceOutput
XformOTMWebServiceOutput_to_ProcessSalesOrderShippingChargeCalculationResponseEBM
ProcessSalesOrderShippingChargeCalculationResponseEBM
ProcessSalesOrderShippingChargeCalculationResponseEBM
ProcessSalesOrderShippingChargeCalculationResponseEBM
ProcessSalesOrderShippingChargeCalculationResponseEBM_to_CalculateShippingChargeSalesOrderSiebelRespMsg
CalculateShippingChargeSalesOrderSiebelRespMsg
CalculateShippingChargeSalesOrderSiebelRespMsg

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 sub tab 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 it 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 it 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.

Data Requirements

Data requirements indicate the mandatory data that you must provide to make the flow successful:

- Order has at least one line item.
- Order line item has a valid product.
- Product weight is specified.
- Source and Target Addresses are specified.
- Source Inventory Location is specified.

Siebel CRM Interfaces

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

- SWIOrderIO.xsd

  Siebel calls CalculateShippingChargeSalesOrderSiebelReqABCSImpl with a CalculateShippingChargeSalesOrderReqMsg ABM.

- SWIOrderIO.xsd

  Siebel expects CalculateShippingChargeSalesOrderRespMsg ABM from CalculateShippingChargeSalesOrderSiebelReqABCSImpl.

For more information about Siebel web services, see the Siebel CRM Integration Pack for Oracle Order Management Addendum

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

Core AIA Components

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

- SalesOrderEBO
- ProcessSalesOrderShippingChargeEBM
- ProcessSalesOrderShippingChargeResponseEBM

The core enterprise business object (EBO) and enterprise business message (EBM) XSD files can be located by EBO within this parent folder: http://[HOST:PORT]/AIAComponents/EnterpriseObjectLibrary/Core/EBO/.

The core EBS WSDL files can be located by EBO within this parent folder: http://[HOST:PORT]/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/.

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

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

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

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

Integration Services

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

- SalesOrderEBS
- CalculateShippingChargeSalesOrderSiebelReqABCSImpl
- ProcessSalesOrderShippingChargeLogisticsProvABCSImpl

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

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

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 Application Integration Architecture - Foundation Pack: Integration Developer's Guide*, "Designing and Developing EBSs" and *Oracle Application Integration Architecture - Foundation Pack: Concepts and Technologies Guide*, "Understanding EBSs"

CalculateShippingChargeSalesOrderSiebelReqABCSImpl

The CalculateShippingChargeSalesOrderSiebelReqABCSImpl is the Application Business Connector (ABC) Service that exposes the following operations related to Shipping Charges integration on the Siebel ABM.

The ProcessSalesOrderShippingChargeCalculation integration flow uses this BPEL process. The CalculateShippingChargeSalesOrderSiebelReqABCSImpl transforms the Siebel Request Message into the ProcessSalesOrderShippingChargeCalculationEBM and invokes the SalesOrderEBS with the operation ProcessSalesOrderShippingCharge.

The SalesOrderEBS routes it 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 it 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

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/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.

Chapter 9: Understanding the Credit Check Integration Flow

This chapter discusses:

- Credit Check requests
- Data requirements
- Siebel Customer Relationship Management (CRM) interfaces
- Oracle E-Business Suite (Oracle EBS) interfaces
- Core Oracle Application Integration Architecture (AIA) components
- Integration services

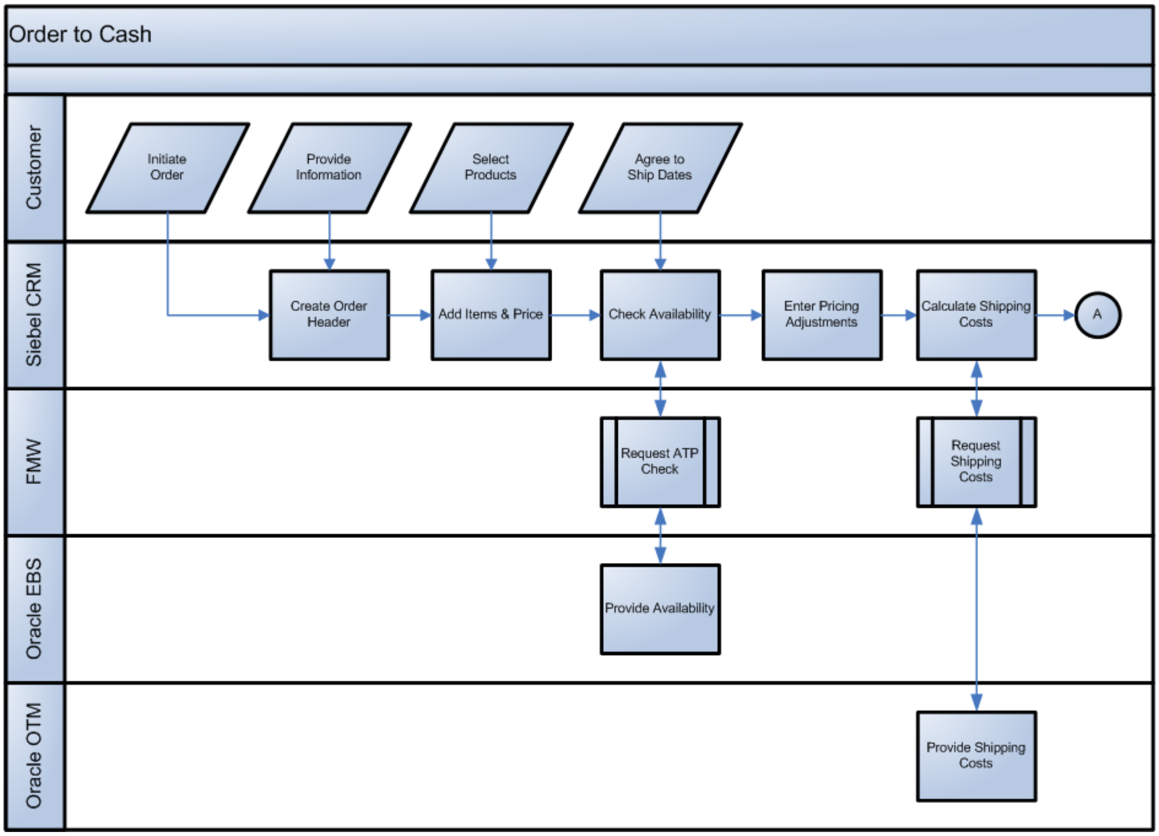
Credit Check Requests

If the order payment method is Purchase Order, then you can call the Credit Check integration flow on request before calling the Create Order integration flow. 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 if the order needs to be on credit hold. Oracle EBS responds with a message to either put the order on hold or not. This data is returned to Siebel CRM so that a Customer Service Representative (CSR) can inform the customer. The order will not be sent to the back office until the hold is removed in Siebel CRM and the order is resubmitted.

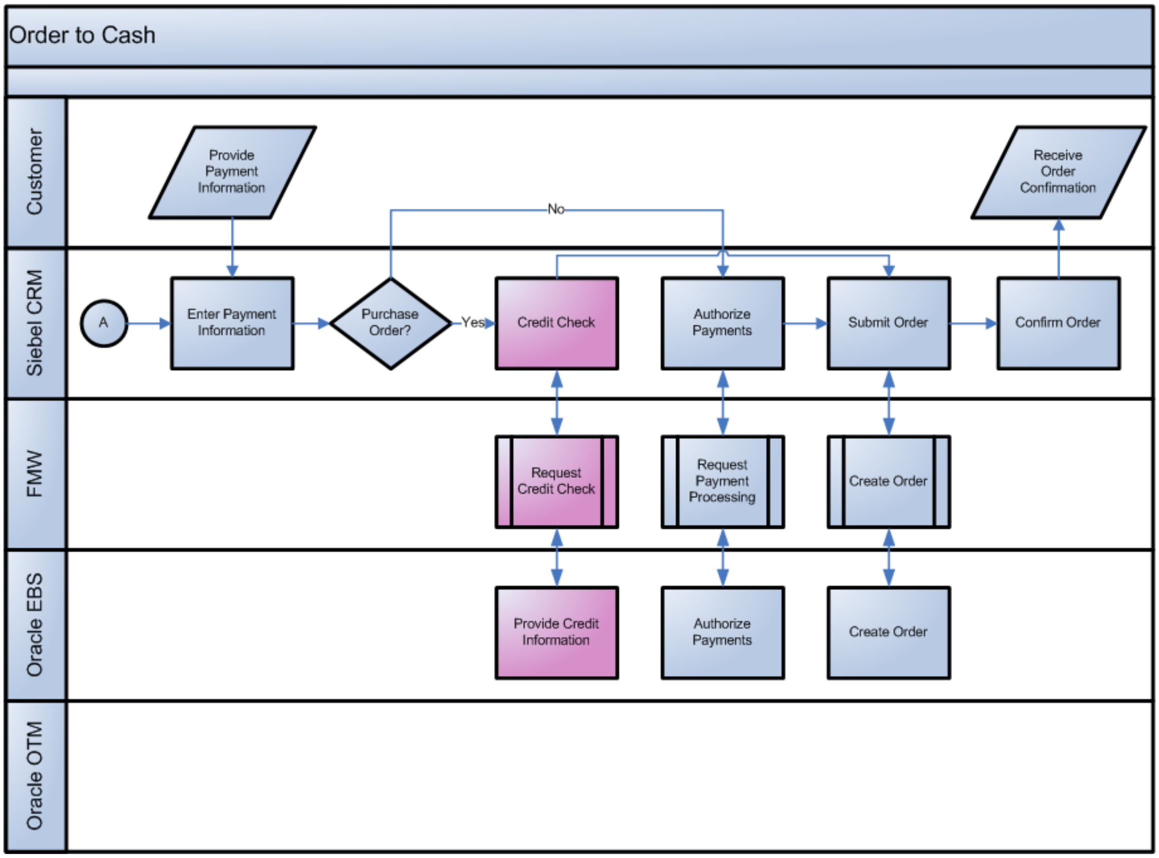
If the credit check fails, then the hold flag is set to true and the purchase order payment status is set to Rejected. When an order is submitted to Oracle EBS, if the hold flag is enabled or if the purchase order status is not set to Approved, then the order is not submitted to the back office system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

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

These diagrams illustrate where the Credit Check integration flow fits in the Order to Cash: Siebel CRM – EBS integration flow:



Credit Check integration flow (1 of 2)



Credit Check integration flow (2 of 2)

Prerequisites

Product synchronization is the prerequisite for the Credit Check integration flow.

For more information about product synchronization, see Chapter 4: Describing the Process Integration for Product Management

Solution Assumptions and Constraints

The assumptions for the Credit Check integration flow are:

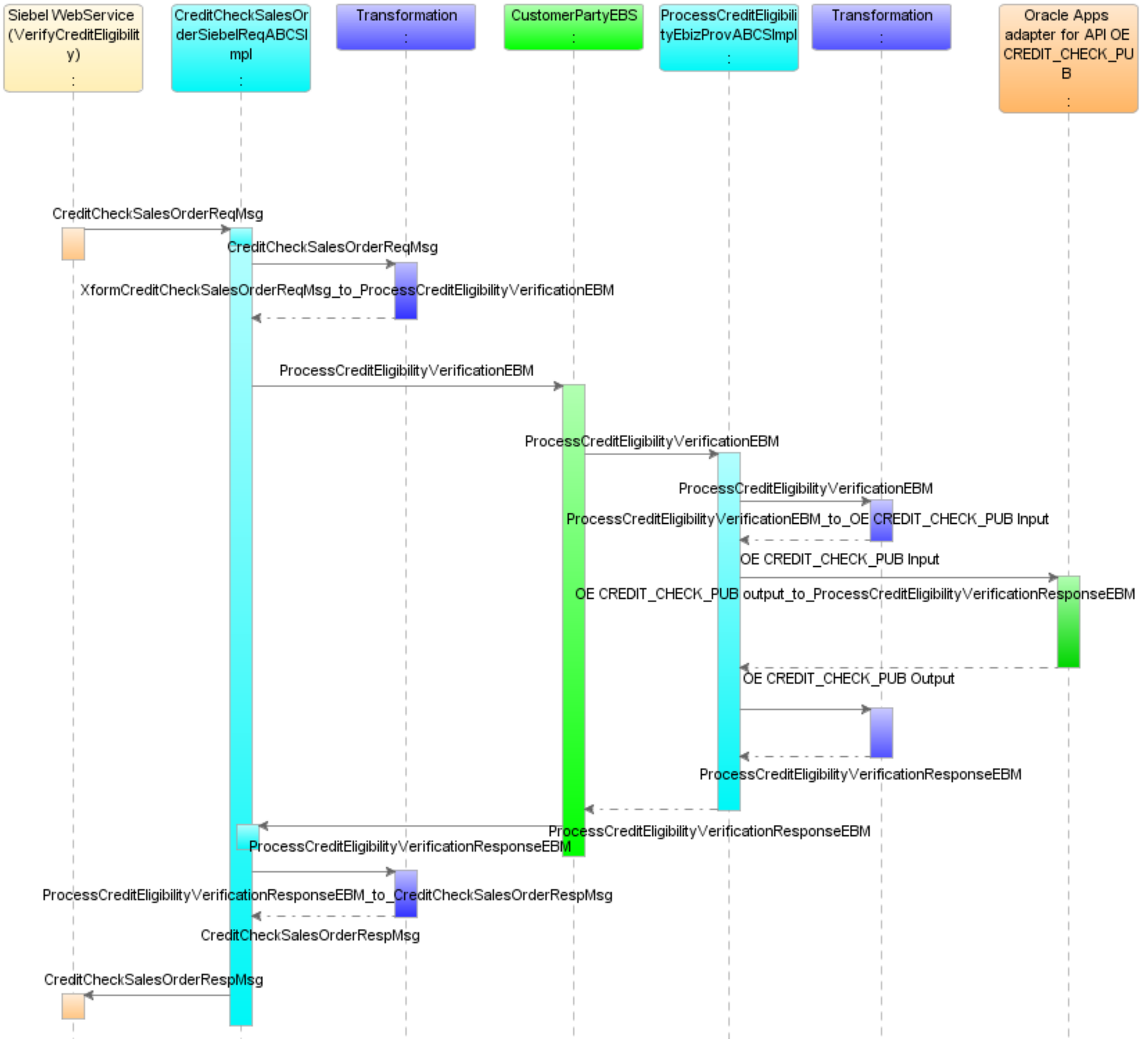
1. The credit profile is maintained in Oracle EBS prior to invoking the credit check.
2. The order payment method is Purchase Order.
3. Credit check is not supported for Siebel quotes.

Credit Check Integration Flow

This integration flow uses the following interfaces:

- CustomerPartyEBS
- CreditCheckSalesOrderSiebelReqABCSImpl
- ProcessCreditEligibilityEbizProvABCSImpl

This sequence diagram illustrates the Credit Check integration flow:


Siebel WebService (VerifyCreditEligibility)
CreditCheckSalesOrderSiebelReqABCSImpl
Transformation
CustomerPartyEBS
ProcessCreditEligibilityEbizProvABCSImpl
Transformation
Oracle Apps adapter for API OE CREDIT_CHECK_PUB
CreditCheckSalesOrderReqMsg
CreditCheckSalesOrderReqMsg
XformCreditCheckSalesOrderReqMsg_to_ProcessCreditEligibilityVerificationEBM
ProcessCreditEligibilityVerificationEBM
ProcessCreditEligibilityVerificationEBM
ProcessCreditEligibilityVerificationEBM
ProcessCreditEligibilityVerificationEBM_to_OE CREDIT_CHECK_PUB Input
OE CREDIT_CHECK_PUB Input
OE CREDIT_CHECK_PUB output_to_ProcessCreditEligibilityVerificationResponseEBM
OE CREDIT_CHECK_PUB Output
ProcessCreditEligibilityVerificationResponseEBM
ProcessCreditEligibilityVerificationResponseEBM
ProcessCreditEligibilityVerificationResponseEBM
ProcessCreditEligibilityVerificationResponseEBM_to_CreditCheckSalesOrderRespMsg
CreditCheckSalesOrderRespMsg
CreditCheckSalesOrderRespMsg

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 and select Purchase Order. Click out of 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. 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 it 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 it 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 that it is approved.

Data Requirements

Data requirements indicate the mandatory data that you must provide to make the 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 Qty is greater than zero.

Siebel CRM Interfaces

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

- SWIOrderIO.xsd

  Siebel calls the CreditCheckSalesOrderSiebelReqABCSImpl with a CreditCheckSalesOrderReqMsg ABM.

- SWIOrderIO.xsd

  Siebel expects the CreditCheckSalesOrderRespMsg ABM from CreditCheckSalesOrderSiebelReqABCSImpl.

For more information about Siebel web services, see the Siebel CRM Integration Pack for Oracle Order Management Addendum

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 EBS web services, see the following Oracle EBS references:
E-Business Suite Electronic Technical Reference Manual (eTRM) located on My Oracle Support under the E-Business Suite Information Center,
Oracle Integration Repository located at http://irep.oracle.com, Oracle Applications Release 11.5.10+ Online Documentation Library, located on the Oracle Technology Network (http://www.oracle.com/technology/documentation/applications.html)

Core AIA Components

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

- CreditEligibility EBO
- ProcessCreditEligibilityVerificationEBM
- ProcessCreditEligibilityVerificationResponseEBM

The core EBO and EBM XSD files can be located by EBO within this parent folder: http://[HOST:PORT]/AIAComponents/EnterpriseObjectLibrary/Core/EBO/.

The core EBS WSDL files can be located by EBO within this parent folder: http://[HOST:PORT]/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/.

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

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

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

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

Integration Services

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

- CustomerPartyEBS
- CreditCheckSalesOrderSiebelReqABCSImpl
- ProcessCreditEligibilityEbizProvABCSImpl

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

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

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 Application Integration Architecture - Foundation Pack: Integration Developer's Guide*, "Designing and Developing EBSs" and *Oracle Application Integration Architecture - Foundation Pack: Concepts and Technologies Guide*, "Understanding EBSs"

CreditCheckSalesOrderSiebelReqABCSImpl

The CreditCheckSalesOrderSiebelReqABCSImpl is the Application Business Connector (ABC) Service that exposes the Credit Check operation related to the Credit Check integration on the Siebel ABM.

This 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 it 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

ProcessCreditEligibilityEbizProvABCSImpl

The ProcessCreditEligibilityEbizProvABCSImpl BPEL process is used by the ProcessCreditEligibilityVerification integration flow. The CustomerPartyEBS routes 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 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.

Chapter 10: Understanding the Payment Authorization Integration Flow

This chapter discusses:

- Payment Authorization requests
- Data requirements
- Siebel Customer Relationship Management (CRM) interfaces
- Oracle E-Business Suite (Oracle EBS) interfaces
- Core Oracle Application Integration Architecture (AIA) components
- Integration services

Payment Authorization Requests

If the payment method is credit card, then the Payment Authorization integration flow can be called on request before calling the Create Order integration flow.

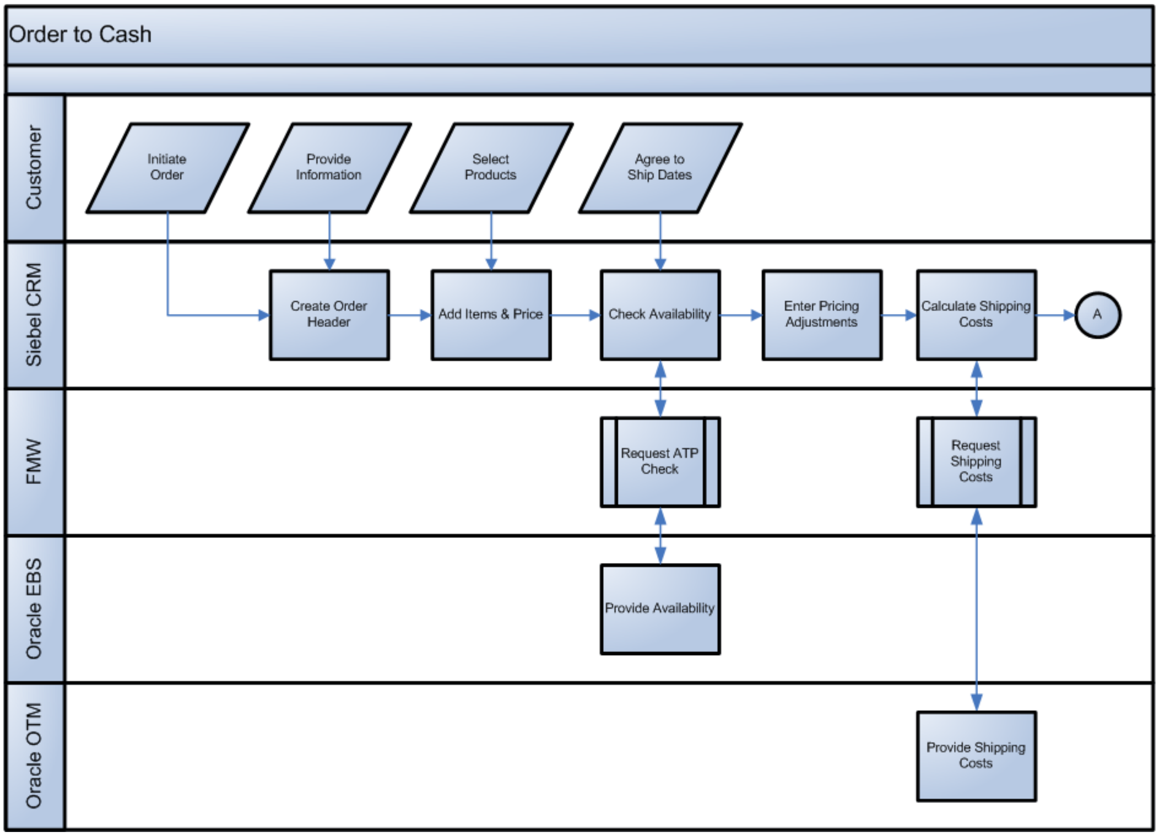
The Payment Authorization integration flow is a synchronous call, initiated in Siebel CRM, which obtains the credit authorization status from Oracle EBS. The credit card details and order amount are passed to Oracle EBS to authorize the payment for the order amount. Oracle EBS responds with a message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

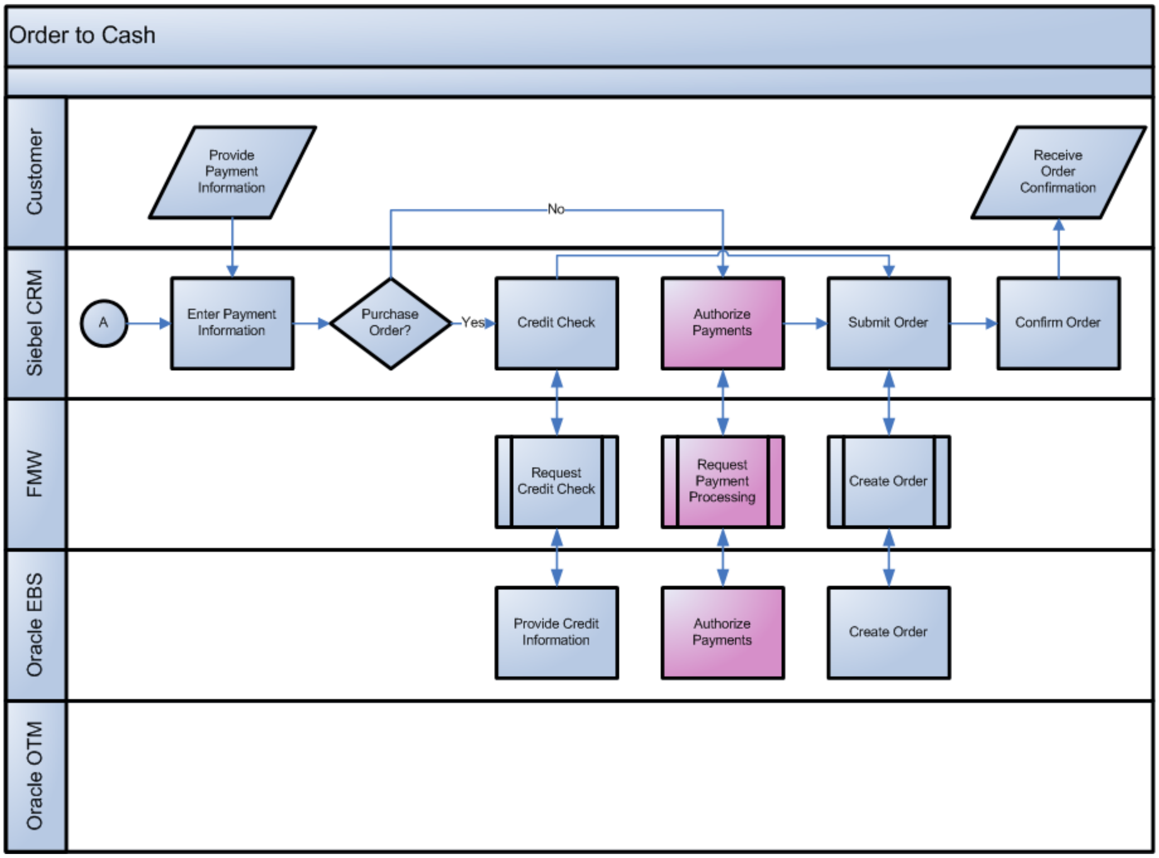
This integration flow works with Siebel sales orders only, not quotes.

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

These diagrams illustrate where the Payment Authorization integration flow fits in the Order to Cash: Siebel CRM – EBS integration flow:



Payment Authorization integration flow (1 of 2)



Payment Authorization Integration Flow (2 of 2)

Prerequisites

For the Payment Authorization flow to work, the iPayment application must be configured properly.

For more information about iPayment, see that product documentation

Solution Assumptions and Constraints

The assumptions for the Payment Authorization integration flow are:

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. Thus, the MM/YYYY date from Siebel CRM is formatted to YYYY-MM-DD 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 will not be authenticated in cases where the current month/year is same as the expiry month/year and current date is beyond the first (1st) 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 12.1.1 due to inherent changes in the iPayment module. If implementations on Oracle EBS 12.1.1 require Credit Card mode of payment, the following options are recommended:

   a 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 Siebel eSales Administration Guide, "Integrating Siebel eSales with Third-Party Payment Applications. “

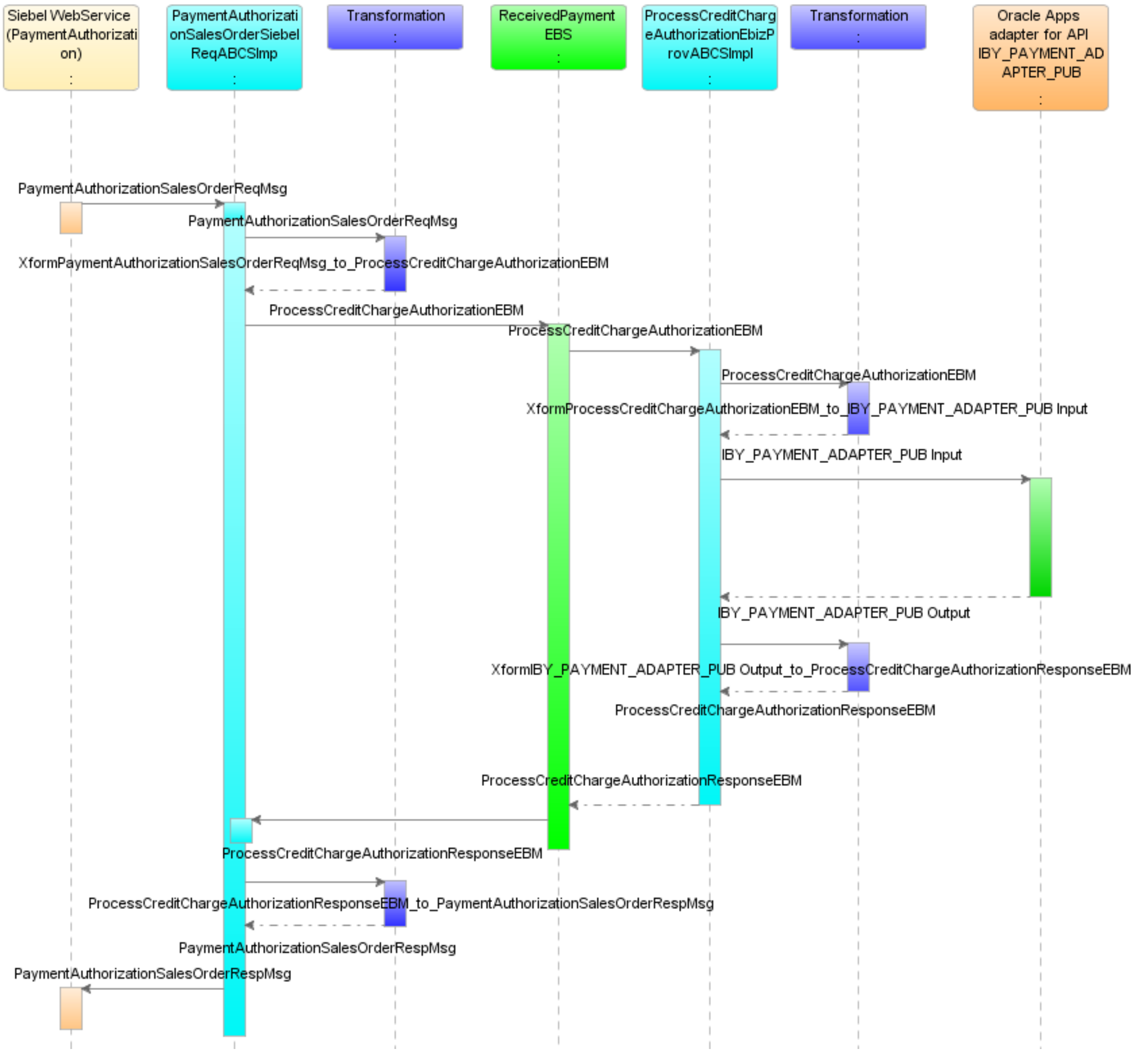
   b Alternately, only the credit card details can be synchronized to Oracle EBS as part of Order sync, and the payment authorization can be accomplished in Oracle EBS itself.

Payment Authorization Integration Flow

This integration flow uses the following interfaces:

- ReceivedPaymentEBS
- PaymentAuthorizationSalesOrderSiebelReqABCSImpl
- ProcessCreditChargeAuthorizationEbizProvABCSImpl

This sequence diagram illustrates the Payment Authorization integration flow:


Siebel WebService (PaymentAuthorization)
PaymentAuthorizationSalesOrderSiebelReqABCSImp
Transformation
ReceivedPayment EBS
ProcessCreditChargeAuthorizationEbizProvABCSImpl
Transformation
Oracle Apps adapter for API IBY_PAYMENT_ADAPTER_PUB
PaymentAuthorizationSalesOrderReqMsg
PaymentAuthorizationSalesOrderReqMsg
XformPaymentAuthorizationSalesOrderReqMsg_to_ProcessCreditChargeAuthorizationEBM
ProcessCreditChargeAuthorizationEBM
ProcessCreditChargeAuthorizationEBM
ProcessCreditChargeAuthorizationEBM
XformProcessCreditChargeAuthorizationEBM_to_IBY_PAYMENT_ADAPTER_PUB Input
IBY_PAYMENT_ADAPTER_PUB Input
IBY_PAYMENT_ADAPTER_PUB Output
XformIBY_PAYMENT_ADAPTER_PUB Output_to_ProcessCreditChargeAuthorizationResponseEBM
ProcessCreditChargeAuthorizationResponseEBM
ProcessCreditChargeAuthorizationResponseEBM
ProcessCreditChargeAuthorizationResponseEBM
ProcessCreditChargeAuthorizationResponseEBM_to_PaymentAuthorizationSalesOrderRespMsg
PaymentAuthorizationSalesOrderRespMsg
PaymentAuthorizationSalesOrderRespMsg

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 it 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 by the.

   The same EBM is returned to the ReceivedPaymentEBS.

7. The ESB service ReceivedPaymentEBS with the message ProcessCreditChargeAuthorizationResponseEBM and the operation ProcessCreditChargeAuthorization routes it 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 successful, the authorization code and date appear in Siebel CRM.

Data Requirements

Data requirements indicate the mandatory data that you must provide 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

Siebel CRM Interfaces

The Siebel interfaces for the Payment Authorization integration flow are:

- SWIOrderIO.xsd

  Siebel should call PaymentAuthorizationSalesOrderSiebelReqABCSImpl with a PaymentAuthorizationSalesOrderReqMsg ABM.

- SWIOrderIO.xsd

  Siebel expects PaymentAuthorizationSalesOrderRespMsg ABM from PaymentAuthorizationSalesOrderSiebelReqABCSImpl.

For more information about Siebel web services, see the Siebel CRM Integration Pack for Oracle Order Management Addendum

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 EBS web services, see the following Oracle EBS references:
E-Business Suite Electronic Technical Reference Manual (eTRM) located on My Oracle Support under the E-Business Suite Information Center,
Oracle Integration Repository located at http://irep.oracle.com, Oracle Applications Release 11.5.10+ Online Documentation Library, located on the Oracle Technology Network (http://www.oracle.com/technology/documentation/applications.html)

Core AIA Components

The Payment Authorization integration flow uses the following horizontal components:

- ReceivedPaymentEBO
- ReceivedPaymentEBS
- ProcessCreditChargeAuthorizationEBM
- ProcessCreditChargeAuthorizationResponseEBM

The core EBO and EBM XSD files can be located by EBO within this parent folder: http://[HOST:PORT]/AIAComponents/EnterpriseObjectLibrary/Core/EBO/.

The core EBS WSDL files can be located by EBO within this parent folder: http://[HOST:PORT]/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/.

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

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

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

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

Integration Services

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

- ReceivedPaymentEBS
- PaymentAuthorizationSalesOrderSiebelReqABCSImpl
- ProcessCreditChargeAuthorizationEbizProvABCSImpl

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

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

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 EBS, see *Oracle Application Integration Architecture - Foundation Pack: Integration Developer's Guide*, "Designing and Developing EBSs" and *Oracle Application Integration Architecture - Foundation Pack: Concepts and Technologies Guide*, "Understanding EBSs"

PaymentAuthorizationSalesOrderSiebelReqABCSImpl

The PaymentAuthorizationSalesOrderSiebelReqABCSImpl is the Application Business Connector (ABC) Service that exposes the following operations related to Payment Authorization integration on the Siebel ABM.

This 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 it 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 it 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

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 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 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

Chapter 11: Understanding Process Integration for Order Management

This chapter discusses:

- Process integration for Order Management
- Sales order creation
- Sales order updates (Oracle EBS initiated)
- Sales order revisions (Siebel CRM initiated)
- Sales order cancellation
- Siebel Customer Relationship Management (CRM) interfaces
- Oracle E-Business Suite (Oracle EBS) interfaces
- Core Oracle Application Integration Architecture (AIA) components
- Integration services

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** enables the synchronization of orders from Siebel CRM to Oracle EBS.
- **Revise sales order** (Siebel CRM initiated) enables the synchronization of order revisions/changes from Siebel CRM to Oracle EBS.
- **Cancel sales orders** enables synchronization of order cancellations from Siebel CRM to Oracle EBS.
- **Update sales orders** (Oracle EBS initiated) enables the synchronization of order updates from Oracle EBS to Siebel CRM.

  In this flow, 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) check service, initiated from Siebel CRM, provides information about when a product can be fulfilled from Oracle EBS.
- **Credit Check** service, initiated from Siebel CRM, obtains credit status information for

purchase orders from Oracle EBS.

- **Payment Authorization** service, initiated in Siebel CRM, obtains the authorization status for credit cards from iPayment.
- **Shipping Charges** service, initiated in Siebel CRM, obtains the estimated shipping cost from Oracle Transportation Management (OTM), formerly GLOG.

Prerequisites

The prerequisites for the Order Management integration are:

- Product synchronization
- Account synchronization
- Price List synchronization

For more information about these process integrations, see Chapter 4: Describing the Process Integration for Product Management, Chapter 3: Describing the Process Integration for Customer Management, and Chapter 5: Describing the Process Integration for Price Lists

Solution Assumptions and Constraints

These are the assumptions and constraints for Order Management:

1. Once products and bills of material (BOMs) are synchronized to Siebel CRM, they cannot be modified in Siebel CRM.
2. Orders are submitted with a status of Booked.
3. Business to Customer (B2C) scenarios are not supported.
4. Return Material Authorization (RMA) is not supported for ATO & PTO items.
5. The ATP check is not supported for ATO and PTO items.
6. When an order is put on hold due to credit status, orders can be submitted only after the credit hold is released.
7. In Oracle EBS there can be multiple holds at the order header and line items. In this solution, only a single header hold is supported.
8. Order updates in Oracle EBS are system-driven. When any of the following attributes change, they are synchronized to Siebel CRM:

   a Order Header/Line Status.

   b Order Header Hold/Hold Release.

   c System-generated line splits due to partial shipping.

   d Line schedule arrival date changes.

   e Line schedule ship date changes.

9. The following Order (or Order line) statuses from Oracle E-Business Suite are supported:

   a Booked

   b Awaiting Shipping

   c Shipped

   d Fulfilled

   e Closed

   f Cancelled

   g Supply Eligible

   Additional statuses can be configured by exposing them in Oracle E-Business Suite, and adding List of Values in Siebel, and DVM in AIA.

10. Orders have a status of Booked before they are submitted. This status is validated by Siebel CRM during the workflow before any integration services are invoked.

11. Siebel CRM allows revisions/changes to an order if the order status is Booked.

12. Customer Account ID, Product ID, and Price List ID must be available before order synchronization is invoked. If they are not synchronized, the processes will error out.

13. When an order with a configured product is synchronized to Oracle EBS, the order gets updated with certain included items/products associated to configured product.

    This order update is synchronized back to Siebel CRM also. 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.

The solution design assumes that the following statements are true:

1. Order validation is outside the scope of the integration flows defined for this release.

   Order validation processes are assumed to have been successfully run before the start of the Create Sales Order integration flow.

2. This integration does not support manual updates to orders in Oracle EBS.

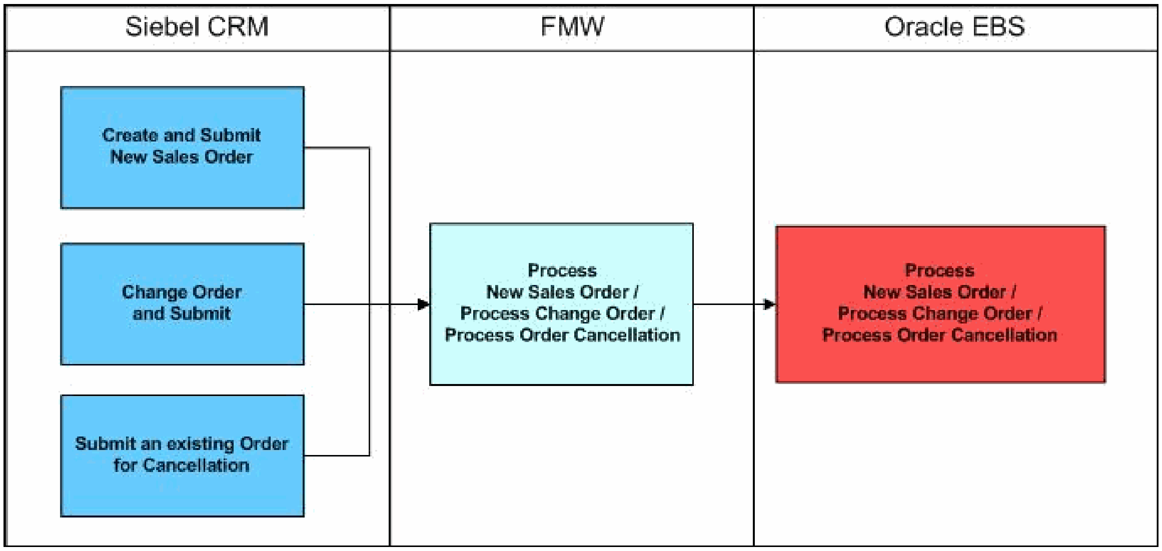
3. Only Siebel orders of the type “Sales Orders” are supported in this integration.

4. Siebel CRM allows revisions/changes to an order if the order status is Booked.

5. The order should be revise-able, if the status of at least one of the order lines is “Booked” or “Awaiting Shipping”.

6. If the status of the line is Shipped, Fulfilled, Closed, Cancelled, Cancel Pending, or Supply Eligible the order line should not be revisable.

This diagram illustrates the overall flow for the process integration:

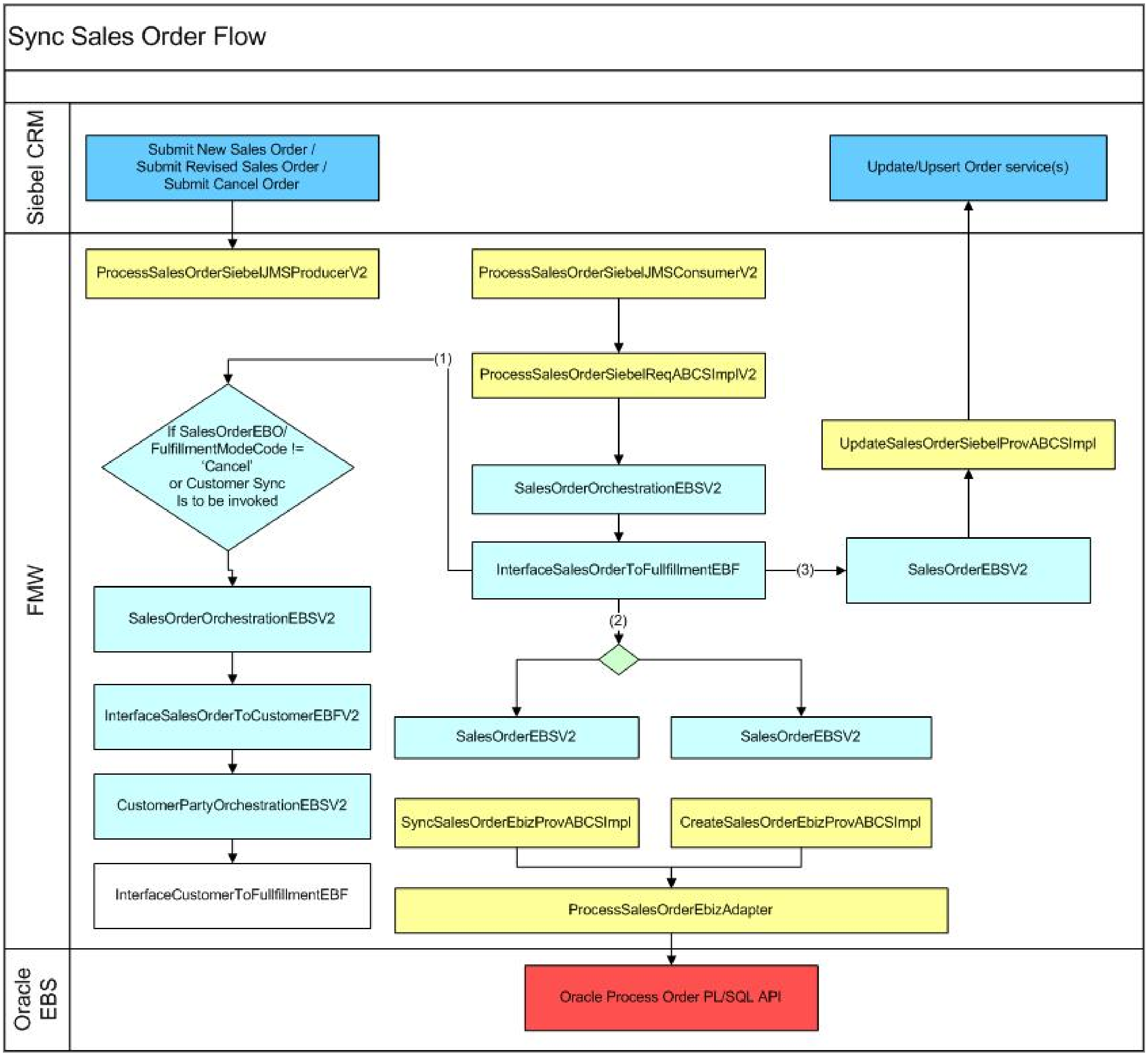

Siebel CRM
FMW
Oracle EBS
Create and Submit New Sales Order
Change Order and Submit
Submit an existing Order for Cancellation
Process New Sales Order / Process Change Order / Process Order Cancellation
Process New Sales Order / Process Change Order / Process Order Cancellation

Order Management overall 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 Fusion Middleware (FMW) sends it to Oracle EBS.
3. Automated order update events occur in Oracle EBS.
4. When a Sales/RMA order update event occurs in Oracle EBS, FMW sends the order update to Siebel CRM.
5. Accounts created in Siebel CRM are not immediately synchronized to Oracle EBS. Instead, accounts are created in Oracle EBS only after an order has been placed in Siebel CRM.
6. 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.
7. The sales order orchestration has two steps. First, SyncCustomer initiates the customer flows. Second, OrderFulfill, which proceeds with order creation in Oracle EBS.

This diagram illustrates the overall Order Management integration flow:


Sync Sales Order Flow
Siebel CRM
FMW
Oracle EBS
Submit New Sales Order / Submit Revised Sales Order / Submit Cancel Order
Update/Upsert Order service(s)
ProcessSalesOrderSiebelJMSProducerV2
ProcessSalesOrderSiebelJMSConsumerV2
(1)
ProcessSalesOrderSiebelReqABCSImplV2
If SalesOrderEBO/ FulfillmentModeCode != 'Cancel' or Customer Sync Is to be invoked
UpdateSalesOrderSiebelProvABCSImpl
SalesOrderOrchestrationEBSV2
InterfaceSalesOrderToFulfillmentEBF
(3)
SalesOrderEBSV2
SalesOrderOrchestrationEBSV2
(2)
InterfaceSalesOrderToCustomerEBFV2
SalesOrderEBSV2
SalesOrderEBSV2
CustomerPartyOrchestrationEBSV2
SyncSalesOrderEbizProvABCSImpl
CreateSalesOrderEbizProvABCSImpl
InterfaceCustomerToFullfillmentEBF
ProcessSalesOrderEbizAdapter
Oracle Process Order PL/SQL API

Overall Order Management integration flow

Sales Order Creation

The Create Sales Order flow enables the synchronization of 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. Prospects are not sent from Siebel CRM to Oracle EBS.

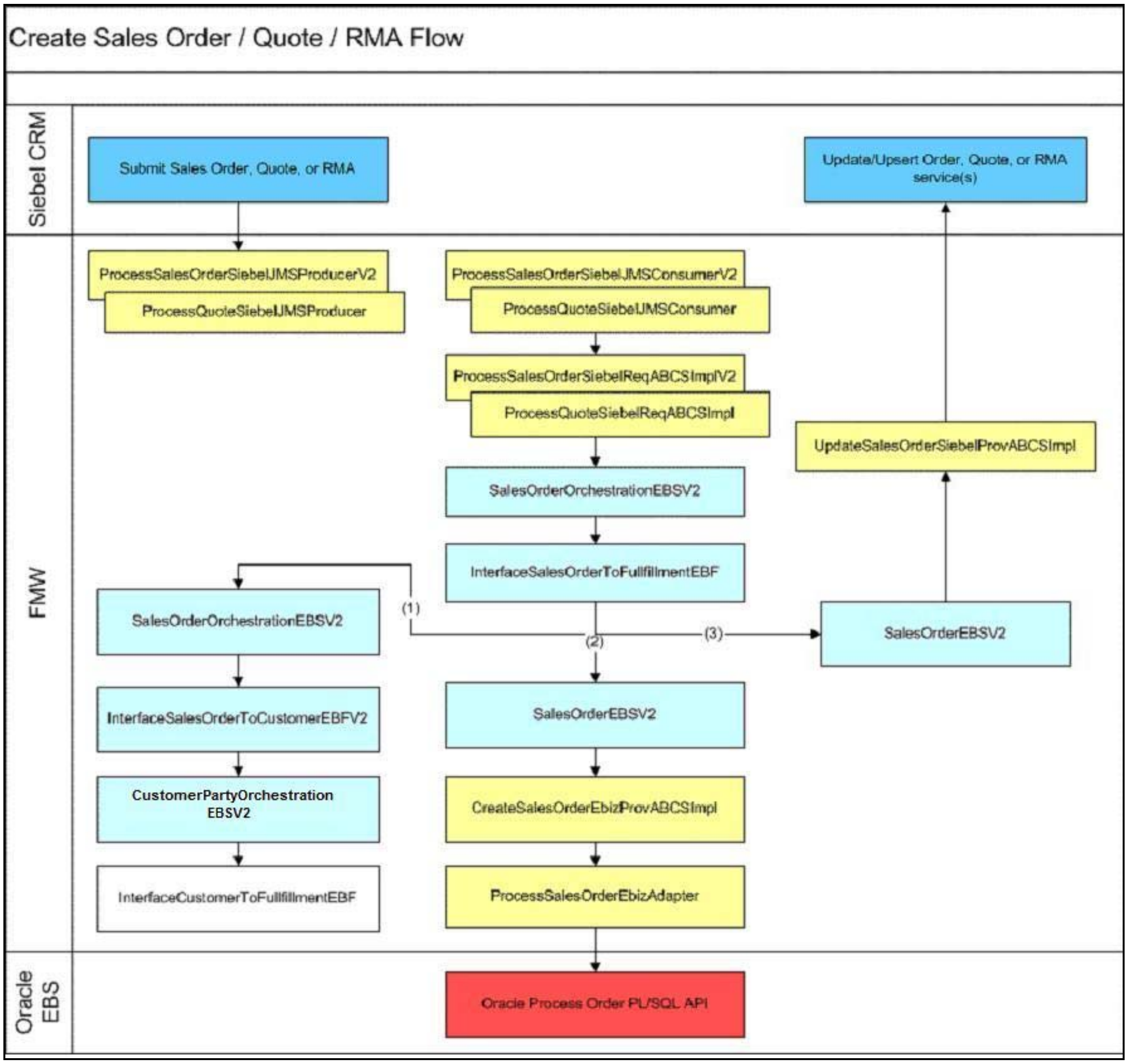
These integration flows are invoked before calling the Create Order process:

- ATP
- Shipping Charges
- Credit Check
- Payment Auth

When a sales order is created 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.

This diagram illustrates the Create Sales Order integration flow:



Create Sales Order / Quote / RMA Flow
Siebel CRM
FMW
Oracle EBS
Submit Sales Order, Quote, or RMA
Update/Upsert Order, Quote, or RMA service(s)
ProcessSalesOrderSiebelJMSProducerV2
ProcessQuoteSiebelJMSProducer
ProcessSalesOrderSiebelJMSConsumerV2
ProcessQuoteSiebelJMSConsumer
ProcessSalesOrderSiebelReqABCSImplV2
ProcessQuoteSiebelReqABCSImpl
UpdateSalesOrderSiebelProvABCSImpl
SalesOrderOrchestrationEBSV2
InterfaceSalesOrderToFullfillmentEBF
(1)
(2)
(3)
SalesOrderOrchestrationEBSV2
SalesOrderEBSV2
SalesOrderEBSV2
InterfaceSalesOrderToCustomerEBFV2
CustomerPartyOrchestration EBSV2
InterfaceCustomerToFullfillmentEBF
CreateSalesOrderEbizProvABCSImpl
ProcessSalesOrderEbizAdapter
Oracle Process Order PL/SQL API

Create Sales Order integration flow

Create Sales Order Integration Flow

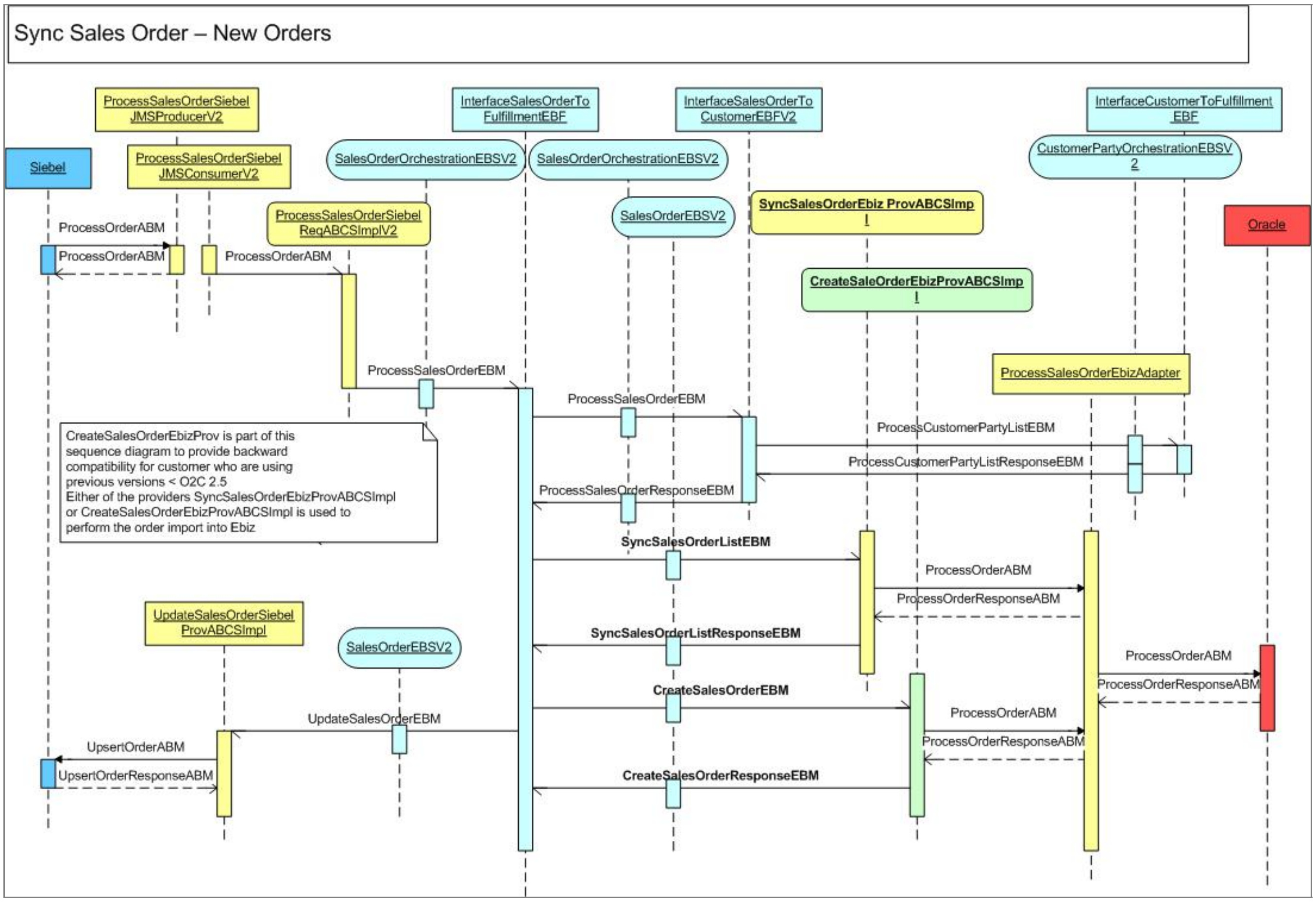
This integration flow uses the following interfaces:

- ProcessSalesOrderSiebelJMSProducerV2
- ProcessSalesOrderSiebelJMSConsumerV2
- ProcessSalesOrderSiebelReqABCSImplV2
- SalesOrderOrchestrationEBSV2
- SalesOrderOrchestrationResponseEBSV2
- InterfaceSalesOrderToFulfillmentEBF
- InterfaceSalesOrderToCustomerEBF
- InterfaceCustomerToFulfillmentEBF
- SalesOrderEBSV2
- CreateSalesOrderEbizProvABCSImpl
- SyncSalesOrderEbizProvABCSImpl
- ProcessSalesOrderEbizAdapter
- SalesOrderResponseEBSV2
- UpdateSalesOrderSiebelProvABCSImpl

The following order attributes are synchronized and passed to Oracle EBS:

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

This sequence diagram illustrates the Create Sales Order integration flow:


Sync Sales Order – New Orders
ProcessSalesOrderSiebel JMSProducerV2
InterfaceSalesOrderTo FulfillmentEBF
InterfaceSalesOrderTo CustomerEBFV2
InterfaceCustomerToFulfillment EBF
Siebel
ProcessSalesOrderSiebel JMSConsumerV2
SalesOrderOrchestrationEBSV2
SalesOrderOrchestrationEBSV2
CustomerPartyOrchestrationEBSV 2
ProcessSalesOrderSiebel ReqABCSImplV2
SalesOrderEBSV2
SyncSalesOrderEbiz ProvABCSImp l
Oracle
CreateSaleOrderEbizProvABCSImp l
ProcessSalesOrderEbizAdapter
ProcessOrderABM
ProcessOrderABM
ProcessOrderABM
ProcessSalesOrderEBM
ProcessSalesOrderEBM
ProcessCustomerPartyListEBM
ProcessCustomerPartyListResponseEBM
ProcessSalesOrderResponseEBM
CreateSalesOrderEbizProv is part of this sequence diagram to provide backward compatibility for customer who are using previous versions < O2C 2.5 Either of the providers SyncSalesOrderEbizProvABCSImpl or CreateSalesOrderEbizProvABCSImpl is used to perform the order import into Ebiz
SyncSalesOrderListEBM
ProcessOrderABM
ProcessOrderResponseABM
UpdateSalesOrderSiebel ProvABCSImpl
SalesOrderEBSV2
SyncSalesOrderListResponseEBM
ProcessOrderABM
ProcessOrderResponseABM
CreateSalesOrderEBM
UpdateSalesOrderEBM
ProcessOrderABM
ProcessOrderResponseABM
UpsertOrderABM
UpsertOrderResponseABM
CreateSalesOrderResponseEBM

Create Sales Order flow sequence diagram

When you initiate the Create Sales Order process, the following events occur:

1. When an Order is created and submitted in Siebel CRM, workflow invokes the ProcessSalesOrderSiebelJMSProducerV2 service.

2. The invoked JMSProducer service enqueues the Siebel ABM to a 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.

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 will be identified as follows:
   If CurrentOrderRowId (SWIOrder/ID) is equal to PreviousOrderId (SWIOrder/PreviousOrderId), the VerbCode is populated as ‘Process’ otherwise the VerbCode will hold ‘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.

   There is a configuration parameter 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 EBM back to the InterfaceSalesOrderToFulfillmentEBF.
10. Upon receiving the response from the InterfaceSalesOrderToCustomerEBF service, if the focus is backward compatibility to support CreateSalesOrderEbizProvABCSImpl for new order creation, this can be controlled by a configuration parameter isLegacyEbizProvSupported and EBMHeader/VerbCode value ‘Process’ (refer 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 will perform 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 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 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 either the UpsertOrder Siebel ABM, using the appropriate cross-reference tables to determine the Siebel IDs from the Common IDs.
18. The UpdateSalesOrderSiebelProvABCSImpl then invokes either the Siebel UpsertOrder web service to update the status of the order header.

Target System Identification and Routing

An appropriate provider ABCS service should be identified, and routed to two points in the Create Order integration flow.

The first point is at the creation of the order in the back office fulfillment system. The SalesOrderEBSV2 CreateSalesOrder operation must invoke the appropriate provider ABC service. When delivered, the target fulfillment system will not be identified until the original CreateSalesOrderEBS routing rules are run and the system determines that the Oracle EBS provider ABC service should be the target. Customers can replace the original routing rules to do more complex target system decision-making and routing.

The second point follows the creation of the order in the back office fulfillment, when the SalesOrderEBSV2 UpdateSalesOrder operation must invoke the provider ABC service 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 ABC service.

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. As 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/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.

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

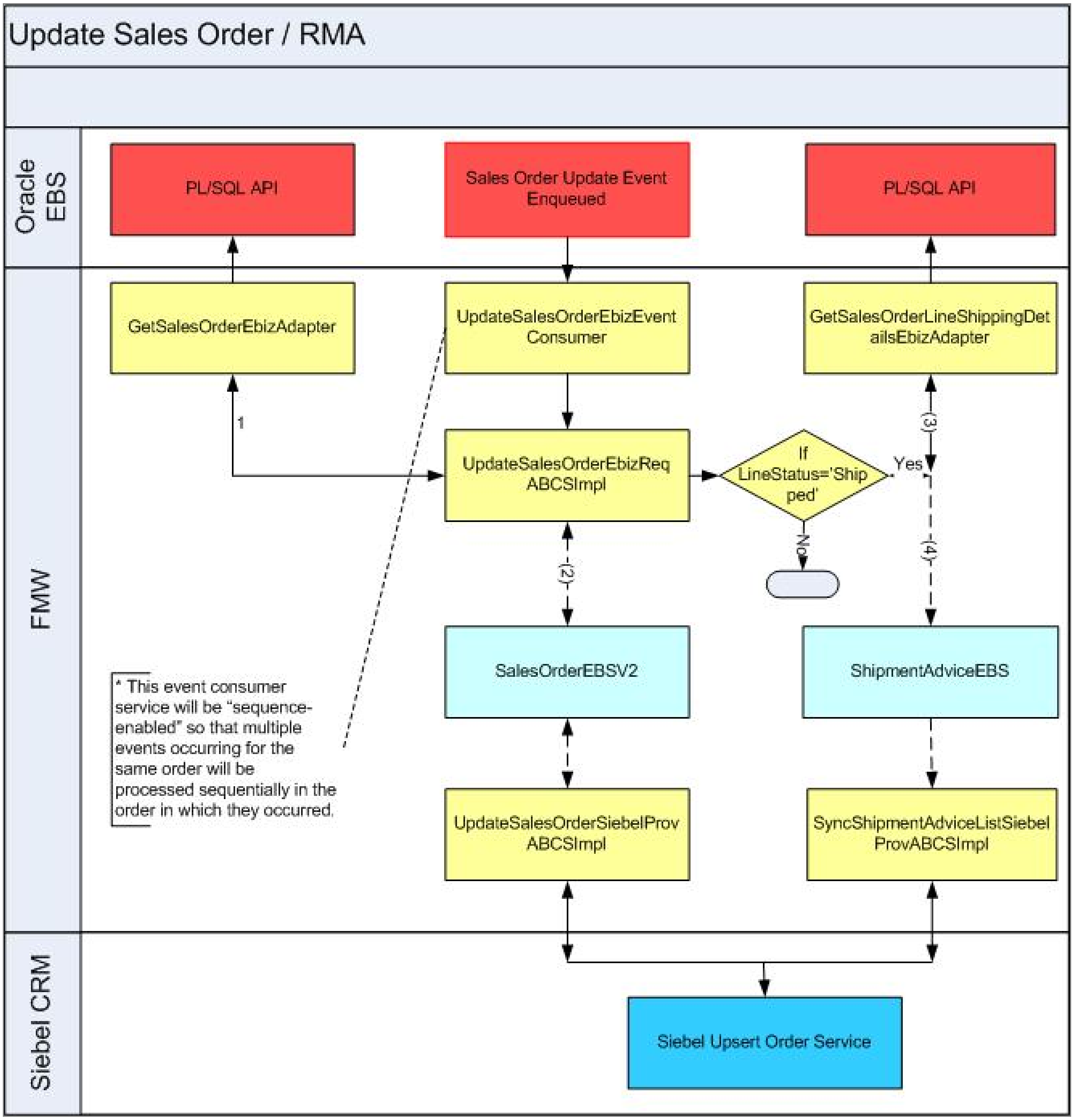
- Order Header/Line Status.

- Order Header Hold/Hold Release.
- System-generated line splits due to partial shipping.
- Line schedule arrival date changes.
- Line schedule ship date changes.
- Order header/line freight, tax charge changes.
- Order line shipping details.

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

This diagram illustrates the Update Sales Order integration flow:


Update Sales Order / RMA
Oracle EBS
FMW
Siebel CRM
PL/SQL API
Sales Order Update Event Enqueued
PL/SQL API
GetSalesOrderEbizAdapter
UpdateSalesOrderEbizEvent Consumer
GetSalesOrderLineShippingDetailsEbizAdapter
1
UpdateSalesOrderEbizReq ABCSImpl
If LineStatus='Shipped'
Yes
No
(3)
(2)
(4)
SalesOrderEBSV2
ShipmentAdviceEBS
* This event consumer service will be "sequence-enabled" so that multiple events occurring for the same order will be processed sequentially in the order in which they occurred.
UpdateSalesOrderSiebelProv ABCSImpl
SyncShipmentAdviceListSiebel ProvABCSImpl
Siebel Upsert Order Service

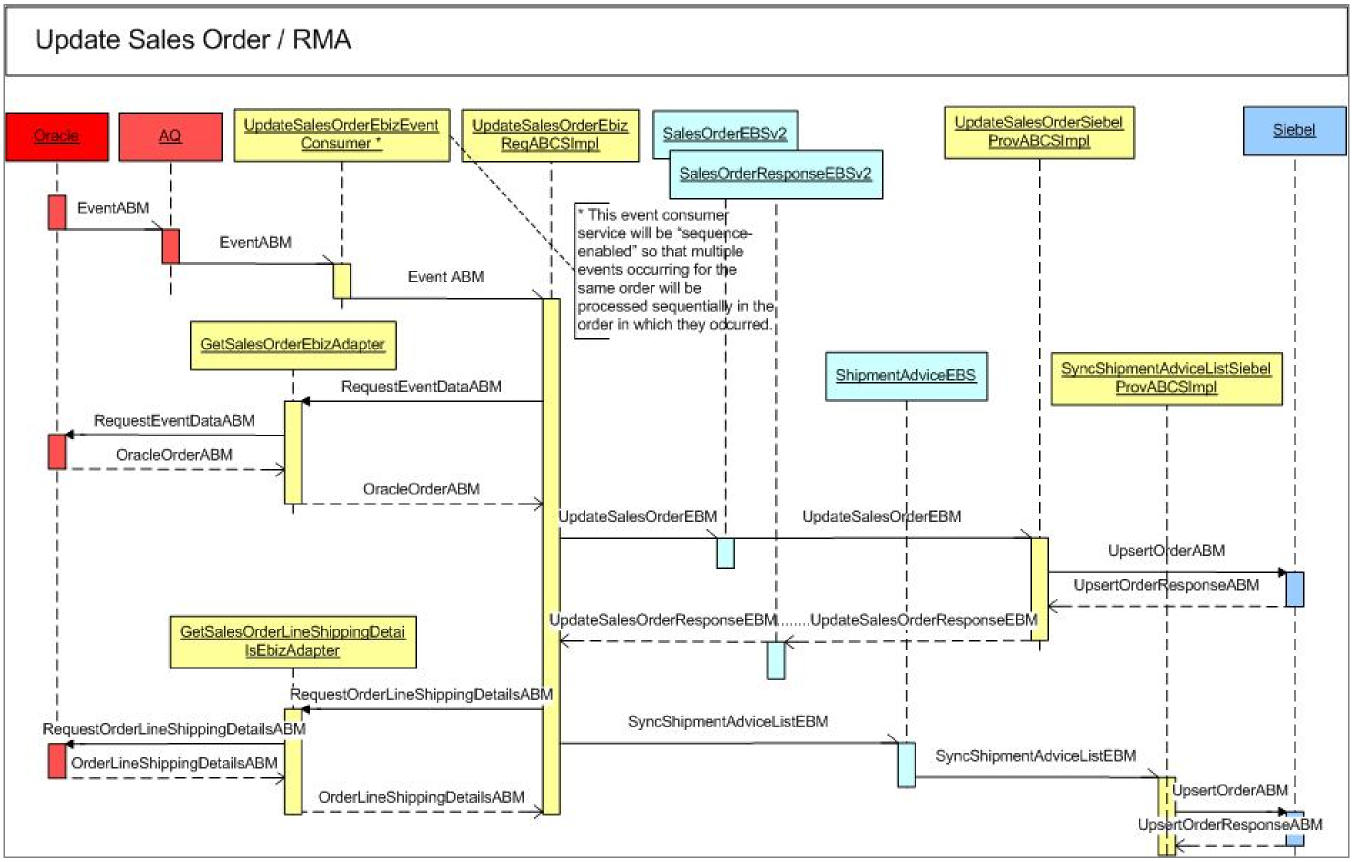
Update Sales Order Integration Flow

This integration flow uses the following interfaces:

- UpdateSalesOrderEbizEventConsumer
- UpdateSalesOrderEbizReqABCSImpl
- GetSalesOrderEbizAdapter

- SalesOrderEBSV2
- SalesOrderResponseEBSV2
- UpdateSalesOrderSiebelProvABCSImpl
- ShipmentAdviceEBS
- ShipmentAdviceResponseEBS
- GetSalesOrderLineShippingDetailsEbizAdapter
- SyncShipmentAdviceListSiebelProvABCSImpl

This sequence diagram illustrates the Update Sales Order integration flow:


Update Sales Order / RMA
Oracle
AQ
UpdateSalesOrderEbizEvent Consumer *
UpdateSalesOrderEbiz ReqABCSImpl
SalesOrderEBSv2
SalesOrderResponseEBSv2
UpdateSalesOrderSiebel ProvABCSImpl
Siebel
EventABM
EventABM
Event ABM
* This event consumer service will be "sequence-enabled" so that multiple events occurring for the same order will be processed sequentially in the order in which they occurred.
GetSalesOrderEbizAdapter
RequestEventDataABM
RequestEventDataABM
OracleOrderABM
OracleOrderABM
ShipmentAdviceEBS
SyncShipmentAdviceListSiebel ProvABCSImpl
UpdateSalesOrderEBM
UpdateSalesOrderEBM
UpsertOrderABM
UpsertOrderResponseABM
UpdateSalesOrderResponseEBM
UpdateSalesOrderResponseEBM
GetSalesOrderLineShippingDetailsEbizAdapter
RequestOrderLineShippingDetailsABM
RequestOrderLineShippingDetailsABM
OrderLineShippingDetailsABM
OrderLineShippingDetailsABM
SyncShipmentAdviceListEBM
SyncShipmentAdviceListEBM
UpsertOrderABM
UpsertOrderResponseABM

Update Sales Order 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 fired, the event is enqueued to the Oracle 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 same order is processed sequentially but the events of different order are processed parallel.

3. The UpdateSalesOrderEbizReqABCSImpl service transforms the EventABM into the Oracle ABM for requesting the Order payload, and then invokes the GetSalesOrderEbizAdapter service for retrieving the full Order update details.
4. The UpdateSalesOrderEbizReqABCSImpl then transforms the full Order update payload into the UpdateSalesOrderEBM to invoke the UpdateSalesOrder operation of the SalesOrderEBSV2.
5. The SalesOrderEBSV2 routes the UpdateSalesOrder operation to the UpdateSalesOrderSiebelProvABCSImpl service.
6. The UpdateSalesOrderSiebelProvABCSImpl service transforms the UpdateSalesOrderEBM into the Upsert Order Siebel ABM, and invokes the Siebel Upsert Order web service.
7. 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.
8. The UpdateSalesOrderSiebelProvABCSImpl service invokes the UpdateSalesOrderResponse operation of SalesOrderResponseEBSV2 which in turn invokes UpdateSalesOrderEbizReqABCSImpl through correlation.
9. The UpdateSalesOrderEbizReqABCSImpl determines If Order payload contains Order line level status equals to ‘shipped’ then it transforms the Oracle Order payload to RequestOrderLineShippingDetailsABM and invokes the GetSalesOrderLineShippingDetailsEbizAdapter for retrieving shipping details for the Order line that has status as ‘shipped’.

   If there is no Order line having ‘shipped’ status then flow will end here.

10. The UpdateSalesOrderEbizReqABCSImpl then transforms OrderLineShippingDetailsABM into SyncShipmentAdviceListEBM to invoke the SyncShipmentAdviceList operation of ShipmentAdviceEBS.
11. The ShipmentAdviceEBS routes the SyncShipmentAdviceList operation to the SyncShipmentAdviceListSiebelProvABCSImpl service.
12. The SyncShipmentAdviceListSiebelProvABCSImpl transforms the SyncShipmentAdviceListEBM into the upsert Order Siebel ABM and invokes the Siebel Upsert Order web service.
13. Upon response from the Siebel web service, the response is transformed into the SyncShipmentAdviceListResponseEBM and then it can invoke SyncShipmentAdviceListResponse operation of ShipmentAdviceResponseEBS.

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 to be generated and sent to AIA for processing. Sequencing of events has been configured in the Update Order flow to provide the following functionality:

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

Target System Determination and Routing

The Update Order flow needs to determine what target system provider ABC service to invoke. As delivered, the UpdateSalesOrderEBS routing rules will 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.

Updating Order Events Sequencing on FMW 10.1.3.4

The following steps must be done manually to enable the re-sequencer functionality in ESB and for better performance. The FMW 10.1.3.4 sequencing feature ensures better scalability and performance. Not defining these does not break anything in the code, but the synchronize Update Order flow will only work as single threaded flow.

To update order events sequencing on FMW 10.1.3.4:

1. Stop SOA server.
2. Open <SOA_Home>/integration/esb/config/esb_config.ini file in an editor.
3. To enable ESB 10.1.3.4 Resequencing functionality add this property in esb_config.ini file - EnableResequencer = true.

   This configuration makes it possible for the routing services to act as a re-sequencer.

4. Configure the following properties in the same esb_config.ini file for better performance of the sequencing feature:

   ResequencerWorkerThreadPoolSize = 5

   ResequencerLockerThreadSleep = 1

   ResequencerMaxRowsRetrieved = 10

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

5. Save and close the file.
6. Restart the SOA server for the changes to take effect.

For more information about using the Oracle ESB Resequencer as a part of an Oracle AIA implementation, see *Oracle Application Integration Architecture – Foundation Pack: Integration Developer's Guide,* "Designing and Constructing ABC Services," Oracle Enterprise Service Bus Resequencer

Sales Order Revision (Siebel CRM Initiated)

All customer requested revisions or changes to the order are done in Siebel CRM. If the order is already booked and submitted to Oracle EBS, these changes need to 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 provided to the user.

An Order in Siebel can be revised only when the status of the order is Booked.

An order line in Siebel can be revised or cancelled 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”, “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”.

Siebel CRM controls the Order or Order line revision using rules in its Data Validation Manager component.

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.

Assumptions and Constraints

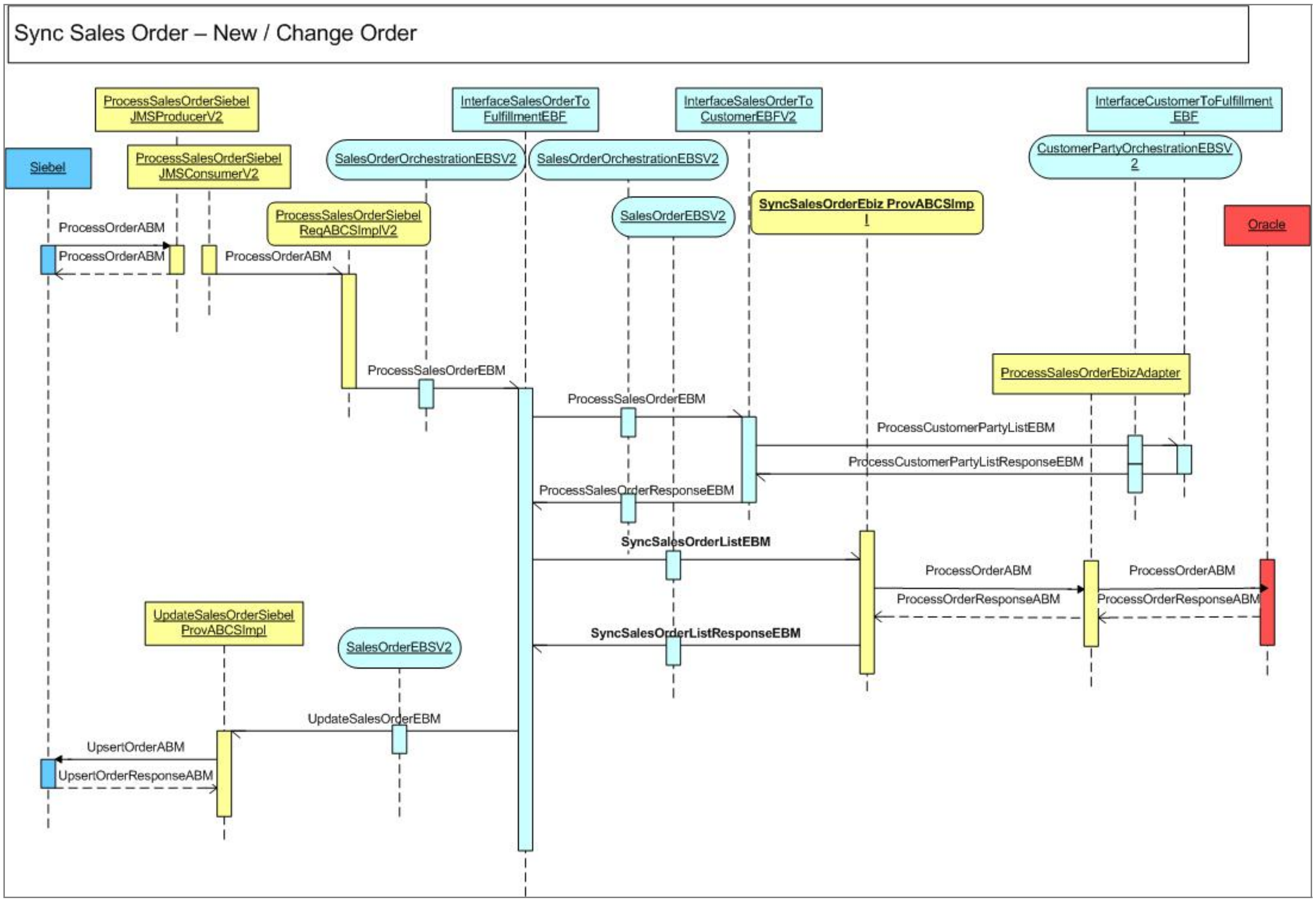
1. If the status of the order is “Booked”, the order can be revised.
2. If the status of an order line is “Booked” or “Awaiting Shipping,” it can be revised. However, the order line cannot be revised if the line status is “Shipped”, ‘Fulfilled”, “Closed”, “Cancelled” “Cancel Pending”, and “Supply Eligible”.
3. Deletion of an existing line in an order in a revision is not supported.

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

This sequence diagram illustrates the Revise Sales Order (Siebel CRM Initiated) integration flow:


Sync Sales Order – New / Change Order
ProcessSalesOrderSiebelJMSProducerV2
InterfaceSalesOrderToFulfillmentEBF
InterfaceSalesOrderToCustomerEBFV2
InterfaceCustomerToFulfillmentEBF
Siebel
ProcessSalesOrderSiebelJMSConsumerV2
SalesOrderOrchestrationEBSV2
SalesOrderOrchestrationEBSV2
CustomerPartyOrchestrationEBSV2
ProcessSalesOrderSiebelReqABCSImplV2
SalesOrderEBSV2
SyncSalesOrderEbiz ProvABCSImpl
Oracle
ProcessOrderABM
ProcessOrderABM
ProcessOrderABM
ProcessSalesOrderEBM
ProcessSalesOrderEbizAdapter
ProcessSalesOrderEBM
ProcessCustomerPartyListEBM
ProcessCustomerPartyListResponseEBM
ProcessSalesOrderResponseEBM
SyncSalesOrderListEBM
ProcessOrderABM
ProcessOrderABM
ProcessOrderResponseABM
ProcessOrderResponseABM
UpdateSalesOrderSiebelProvABCSImpl
SalesOrderEBSV2
SyncSalesOrderListResponseEBM
UpdateSalesOrderEBM
UpsertOrderABM
UpsertOrderResponseABM

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, workflow invokes the ProcessSalesOrderSiebelJMSProducerV2 service.

2. The invoked JMSProducer service enqueues the Siebel ABM to a 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.

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 cancelled 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.

There is a configuration parameter 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/Contact has changed, only then the customer sync should 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 API that results in the updation 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.

Sales Order Cancellation

If any order is cancelled in Order capture system (Siebel CRM), it must be cancelled in the Fulfillment system (Oracle EBS) as well. The order cancellation flow requests to cancel entire order, cancelling all open lines in an order. In this scenario, the order level cancellation is initiated in Siebel CRM through an order revision, to be synched with Oracle EBS. If the fulfillment system (Oracle EBS) rejects the order cancellation for any reason, the order revision in Siebel CRM will have to be rolled back to prior version.

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

Assumptions

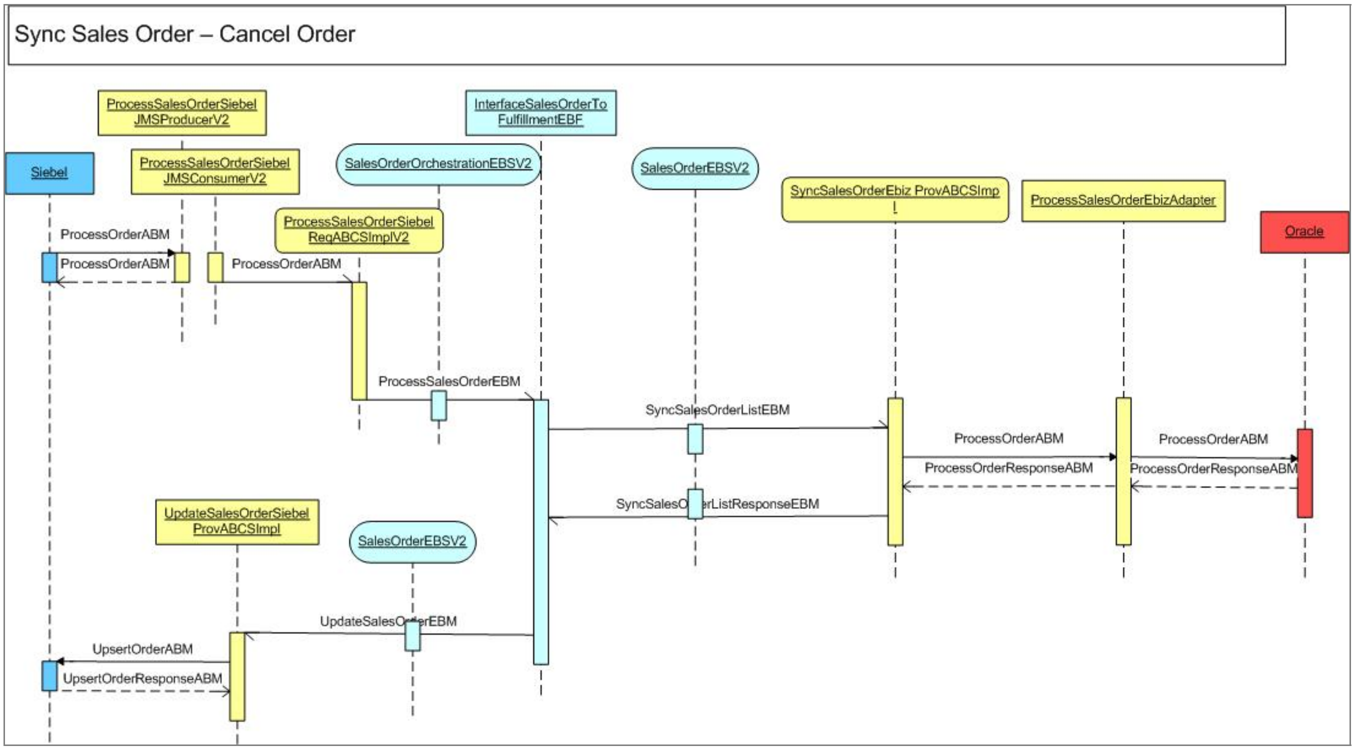
1. If the order status is “Booked” only then the order can be cancelled.
2. Entire order cancellation will not be possible, if any of order line status is “Shipped,” “Fulfilled,” “Closed,” “Cancelled,” “Cancel Pending,” or “Supply Eligible.”

Cancel Sales Orders Integration Flow

This integration flow uses the following interfaces:

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

This sequence diagram illustrates the Cancel Sales Order integration flow:


Sync Sales Order – Cancel Order
ProcessSalesOrderSiebel JMSProducerV2
InterfaceSalesOrderTo FulfillmentEBF
Siebel
ProcessSalesOrderSiebel JMSConsumerV2
SalesOrderOrchestrationEBSV2
SalesOrderEBSV2
SyncSalesOrderEbiz ProvABCSImp l
ProcessSalesOrderEbizAdapter
Oracle
ProcessSalesOrderSiebel ReqABCSImplV2
ProcessOrderABM
ProcessOrderABM
ProcessOrderABM
ProcessSalesOrderEBM
SyncSalesOrderListEBM
ProcessOrderABM
ProcessOrderResponseABM
ProcessOrderABM
ProcessOrderResponseABM
SyncSalesOrderListResponseEBM
UpdateSalesOrderSiebel ProvABCSImpl
SalesOrderEBSV2
UpdateSalesOrderEBM
UpsertOrderABM
UpsertOrderResponseABM

Cancel Sales Order 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, workflow invokes the ProcessSalesOrderSiebelJMSProducerV2 service.
2. The invoked JMSProducer service enqueues the Siebel ABM (holding only header parameters) to a 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.
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 RowIds 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 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.

Siebel CRM Interfaces

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

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 has done in the back office and also to update shipping details of the order line.
- Service Name: QuoteUpsertService

  The UpdateSalesOrderSiebelProvABCSImpl invokes this service during the CreateQuote integration flow after the quote has been synced to 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 Siebel CRM Integration Pack for Oracle Order Management Addendum

Oracle EBS Interfaces

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

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 EBS web services, see the following Oracle EBS references:
E-Business Suite Electronic Technical Reference Manual (eTRM) located on My Oracle Support under the E-Business Suite Information Center,
Oracle Integration Repository located at http://irep.oracle.com,
Oracle Applications Release 11.5.10+ Online Documentation Library, located on the Oracle Technology Network (http://www.oracle.com/technology/documentation/applications.html)

Core 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
- ShipmentAdviceEBO
- SyncShipmentAdviceListEBM
- SyncShipmentAdviceListResponseEBM

The core EBO and EBM XSD files can be located by EBO within this parent folder: http://[HOST:PORT]/AIAComponents/EnterpriseObjectLibrary/Core/EBO/.

The core EBS WSDL files can be located by EBO within this parent folder: http://[HOST:PORT]/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/.

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

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

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

For more information, see *Oracle Application Integration Architecture - Foundation Pack: Integration Developer’s Guide*, “Extensibility for Oracle AIA Artifacts”

Integration Services

The integration provides these services:

- SalesOrderEBSV2
- SalesOrderResponseEBSV2
- SalesOrderOrchestrationEBSV2
- SalesOrderOrchestrationResponseEBSV2
- CustomerPartyOrchestrationEBSV2
- CustomerPartyOrchestrationResponseEBSV2
- InterfaceSalesOrderToFulfillmentEBF
- InterfaceSalesOrderToCustomerEBFV2
- ProcessSalesOrderSiebelJMSProducerV2
- ProcessQuoteSiebelJMSProducer
- ProcessSalesOrderSiebelJMSConsumerV2
- ProcessSalesOrderSoapMsgSiebelJMSConsumer
- ProcessQuoteSiebelJMSConsumer
- ProcessSalesOrderSiebelReqABCSImplV2
- ProcessQuoteSiebelReqABCSImpl
- CreateSalesOrderEbizProvABCSImpl
- UpdateSalesOrderEbizEventConsumer
- UpdateSalesOrderEbizReqABCSImpl
- UpdateSalesOrderSiebelProvABCSImpl
- ProcessSalesOrderEbizAdapter
- GetSalesOrderEbizAdapter
- GetSalesOrderLineShippingDetailsEbizAdapter
- ShipmentAdviceEBS
- ShipmentAdviceResponseEBS
- SyncShipmentAdviceListSiebelProvABCSImpl

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

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

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 service is implemented as ESB routing services.

For more information about this EBS, see *Oracle Application Integration Architecture - Foundation Pack: Integration Developer’s Guide*, “Designing and Developing EBSs” and *Oracle Application Integration Architecture - Foundation Pack: Concepts and Technologies Guide*, “Understanding EBSs”

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

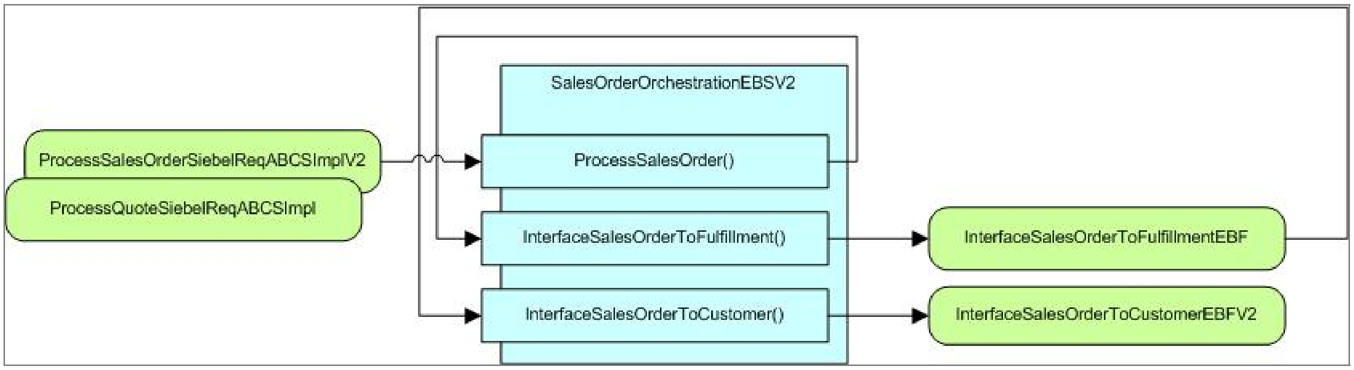
The SalesOrderResponseEBSV2 service is implemented as ESB routing services.

For more information about this EBS, see *Oracle Application Integration Architecture - Foundation Pack: Integration Developer’s Guide*, “Designing and Developing EBSs” and *Oracle Application Integration Architecture - Foundation Pack: Concepts and Technologies Guide*, “Understanding EBSs”

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

(InterfaceSalesOrderToCustomer is need not be invoked for cancel order as order is not revised in this case).


SalesOrderOrchestrationEBSV2
ProcessSalesOrderSiebelReqABCSImplV2
ProcessQuoteSiebelReqABCSImpl
ProcessSalesOrder()
InterfaceSalesOrderToFulfillment()
InterfaceSalesOrderToCustomer()
InterfaceSalesOrderToFulfillmentEBF
InterfaceSalesOrderToCustomerEBFV2

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 routes 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 to the custom orchestration process rather than the InterfaceSalesOrderToFulfillment.
- InterfaceSalesOrderToFulfillment

  This operation routes to the InterfaceSalesOrderToFulfillmentEBF.
- InterfaceSalesOrderToCustomer

  This operation routes to the InterfaceSalesOrderToCustomerEBFV2.

SalesOrderOrchestrationResponseEBSV2 operations:

- ProcessSalesOrderResponse

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

- InterfaceSalesOrderToFulfillmentResponse

  This operation routes to the ProcessSalesOrderResponse operation as indicated in the EBM header.

- InterfaceSalesOrderToCustomerResponse

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

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

CustomerPartyOrchestrationEBSV2

The CustomerPartyOrchestrationEBSV2 service is an EBS that provides 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 *Oracle Application Integration Architecture - Foundation Pack: Integration Developer's Guide*, "Designing and Developing EBSs" and *Oracle Application Integration Architecture - Foundation Pack: Concepts and Technologies Guide*, "Understanding EBSs"

CustomerPartyOrchestrationResponseEBSV2

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

CustomerPartyOrchestrationResponseEBSV2 operations:

InterfaceCustomerToFulfillmentResponse

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

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.

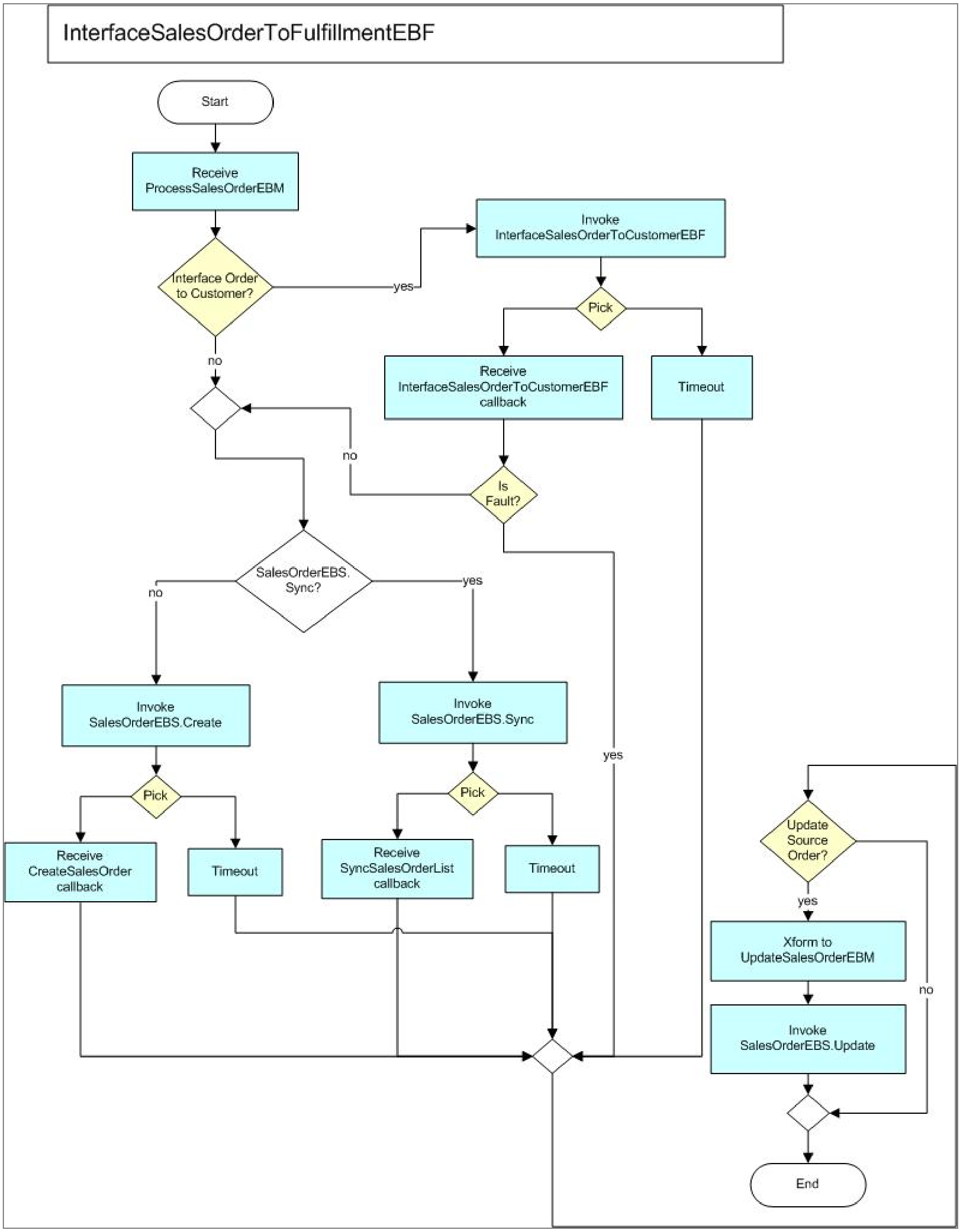
   This step can be configured so that it can be suppressed.

The InterfaceSalesOrderToFulfillmentEBF is an asynchronous BPEL process. Upon completion, it invokes the InterfaceSalesOrderToFulfillmentResponse operation of the SalesOrderOrchestrationResponseEBSV2.

This 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:

- InterfaceSalesOrderToFulfillment
- InterfaceSalesOrderToCustomerResponse
- CreateSalesOrderResponse
- UpdateSalesOrderResponse
- SyncSalesOrderListResponse

This diagram illustrates the InterfaceSalesOrderToFulfillmentEBF activity flow:


InterfaceSalesOrderToFulfillmentEBF
Start
Receive ProcessSalesOrderEBM
Interface Order to Customer?
yes
no
Invoke InterfaceSalesOrderToCustomerEBF
Pick
Receive InterfaceSalesOrderToCustomerEBF callback
Timeout
Is Fault?
no
yes
SalesOrderEBS. Sync?
no
yes
Invoke SalesOrderEBS.Create
Invoke SalesOrderEBS.Sync
Pick
Pick
Receive CreateSalesOrder callback
Timeout
Receive SyncSalesOrderList callback
Timeout
Update Source Order?
yes
no
Xform to UpdateSalesOrderEBM
Invoke SalesOrderEBS.Update
End

InterfaceSalesOrderToFulfillmentEBF activity flow diagram

These are the transformations:

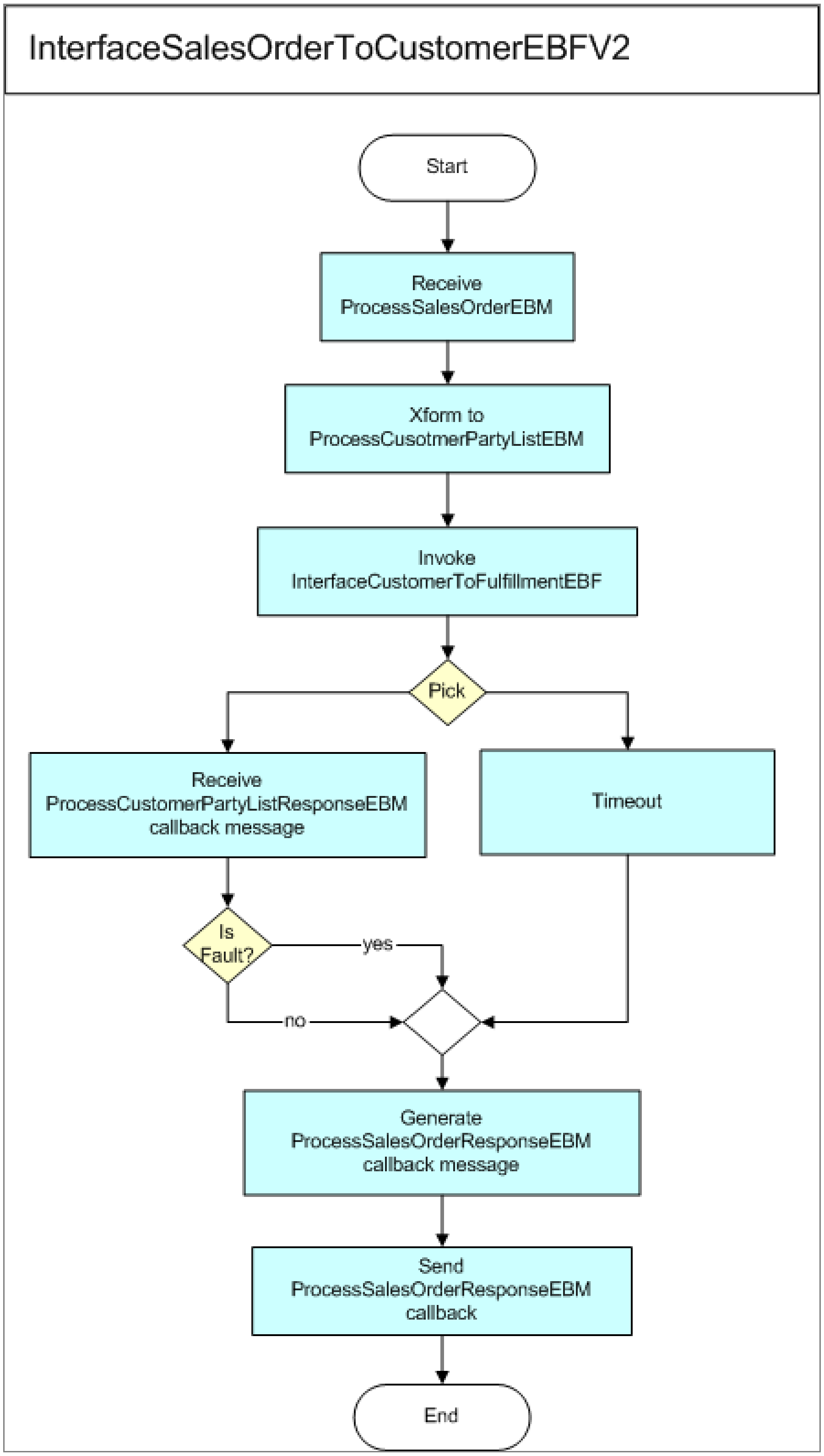
- ProcessSalesOrderEBM to CreateSalesOrderEBM
- ProcessSalesOrderEBM + CreateSalesOrderResponseEBM to UpdateSalesOrderEBM
- ProcessSalesOrderEBM to SyncSalesOrderListEBM
- ProcessSalesOrderEBM + SyncSalesOrderListResponseEBM to UpdateSalesOrderEBM

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.

This diagram illustrates the InterfaceSalesOrderToCustomerEBFV2 activity flow:


InterfaceSalesOrderToCustomerEBFV2
Start
Receive ProcessSalesOrderEBM
Xform to ProcessCusotmerPartyListEBM
Invoke InterfaceCustomerToFulfillmentEBF
Pick
Receive ProcessCustomerPartyListResponseEBM callback message
Timeout
Is Fault?
yes
no
Generate ProcessSalesOrderResponseEBM callback message
Send ProcessSalesOrderResponseEBM callback
End

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.

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.

It is assumed that the Siebel order was already validated from within the Siebel workflow.

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

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 8.0.0.7 or lower.

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 ESB.

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.

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 ESB.

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 populated 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, order lines are re-generated and populated into the cross-reference to point to new Siebel Row IDs and referenced accounts, ship to, bill to addresses, and contacts are populated or looked up from the cross-reference.
- In the cancel order integration flow: As part of the transformation to the ProcessSalesOrderEBM, common IDs for 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.

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 using 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 ESB.

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.

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 ESB.

These are the transformations:

- SyncSalesOrderListEBM to ProcessSalesOrder ABM
- ProcessSalesOrderResponse ABM to SyncSalesOrderListResponseEBM

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

UpdateSalesOrderEbizEventConsumer

The UpdateSalesOrderEbizEventConsumer service subscribes to the oracle.apps.ont.genesis.outbound.update business event in Oracle EBS. Once the event is picked up from the 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 ESB when the subscription event occurs.

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

UpdateSalesOrderEbizReqABCSImpl

The UpdateSalesOrderEbizReqABCSImpl service fetches the Oracle Order ABM from GetSalesOrderEbizAdapter based on event payload and transforms the Oracle order ABM into UpdateSalesOrderresponseEBM and then invokes the UpdateSalesOrder operation of the SalesOrderEBSV2.

After getting UpdateSalesOrderResponsEBM through correlation, if any of Order line status is ‘shipped’ then it will fetch Order Line shipping details ABM and transform OrderLineShippingDetailsABM into the SyncShipmentAdviceListEBM and then invoke SyncShipmentAdviceList operation of ShipmentAdviceEBS.

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

These are the transformations:

- WF_EVENT_T_msg → args_in_msg
- Args_out_msg → UpdateSalesOrderEBM
- Args_out_msg → RequestOrderLineShippingDetails ABM
- OrderLineShippingDetailsABM → SyncShipmentAdviceListEBM

The UpdateSalesOrderEbizReqABCSImpl service is implemented as an asynchronous request-delayed response BPEL process.

UpdateSalesOrderSiebelProvABCSImpl

The UpdateSalesOrderSiebelProvABCSImpl service is the Siebel provider application business connector service 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. Once completed, 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

ProcessSalesOrderEbizAdapter

The ProcessSalesOrderEbizAdapter service is an Oracle EBS Adapter service registered in ESB. 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 ESB, ESB will expose a 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 ESB.

GetSalesOrderEbizAdapter

The GetSalesOrderEbizAdapter service is an Oracle EBS Adapter service registered in ESB. 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 fired 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

GetSalesOrderLineShippingDetailsEbizAdapter

The GetSalesOrderLineShippingDetailsEbizAdapter service is an Oracle EBS Adapter service registered in ESB. 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. By registering this adapter service in ESB, ESB exposes SOAP binding that is used in this integration to invoke the service from the UpdateSalesOrderEbizReqABCSImpl.

ShipmentAdviceEBS

The ShipmentAdviceEBS service is an EBS that provides the basic request operations that can be performed against the ShipmentAdvice EBO. This service gets invoked as part of the Update Order integration flow.

ShipmentAdviceEBS Operation:

- SyncShipmentAdviceList

  The ShipmentAdviceEBS service is implemented as ESB routing services.

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

ShipmentAdviceResponseEBS

The ShipmentAdviceResponseEBS service is an EBS that provides the basic response operations that can be performed against the ShipmentAdvice EBO. This service is invoked as part of Update Order integration flow.

ShipmentAdviceResponseEBS Operation:

- SyncShipmentAdviceListResponse

The ShipmentAdviceResponseEBS service is implemented as ESB routing services.

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

SyncShipmentAdviceListSiebelProvABCSImpl

The SyncShipmentAdviceListSiebelProvABCSImpl service is the Siebel provider application business connector service for the SyncShipmentAdviceList operation of the ShipmentAdviceEBS. This service invokes the Siebel Upsert Order web service.

This service implements the SyncShipmentAdviceList operation defined in the ShipmentAdviceEBS service. This operation is an asynchronous request operation that accepts the SyncShipmentAdviceList EBM. Once complete, this service invokes the SyncShipmentAdviceList Response operation of the ShipmentAdviceEBS.

These are the transformations:

- SyncShipmentAdviceListEBM to UpsertOrderABM
- UpsertOrder Response ABM to SyncShipmentAdviceListResponseEBM

Chapter 12: Understanding Process Integration for Asset Management

This chapter discusses:

- Process integration for Asset Management
- Asset creation
- Asset updates
- Data requirements
- Siebel Customer Relationship Management (CRM) interfaces
- Oracle E-Business Suite (Oracle EBS) interfaces
- Core Oracle Application Integration Architecture (AIA) components
- Integration services

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** enables the synchronization of new customer-owned Item Instances from Oracle EBS to Siebel CRM Asset.
- **Update assets** enables the synchronization of updates to Item Instances from Oracle EBS to Siebel CRM Asset.
- **Bulk load of assets** enables the extraction, transformation, and loading 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 interface tables. This process also enables cross-referencing between Oracle EBS and Siebel CRM.

For more information about initial data loads, see Chapter 2: Describing the Initial Data Loads

Prerequisites

The cross-business process functional prerequisites for asset integration are:

- Product synchronization.

- Customer Account synchronization.

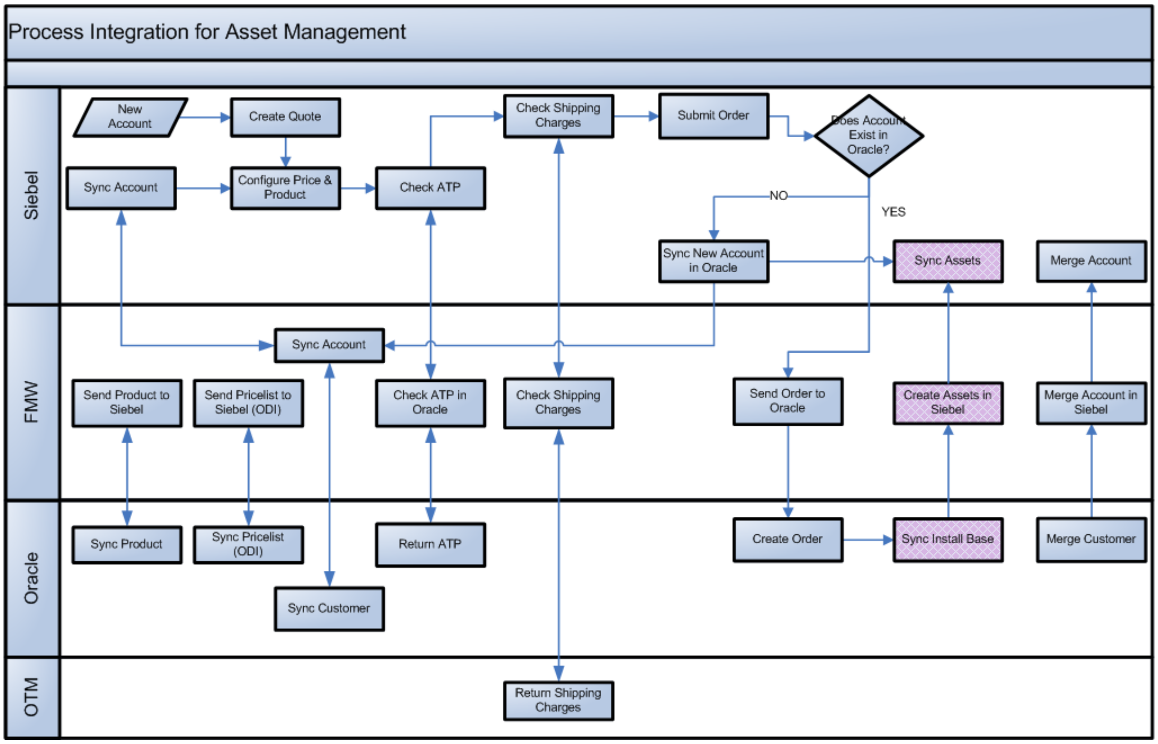
For more information about these process integrations, see Chapter 4: Describing the Process Integration for Product Management, and Chapter 3: Describing the Process Integration for Customer Management

Solution Assumptions and Constraints

These are the assumptions and constraints for the asset integration:

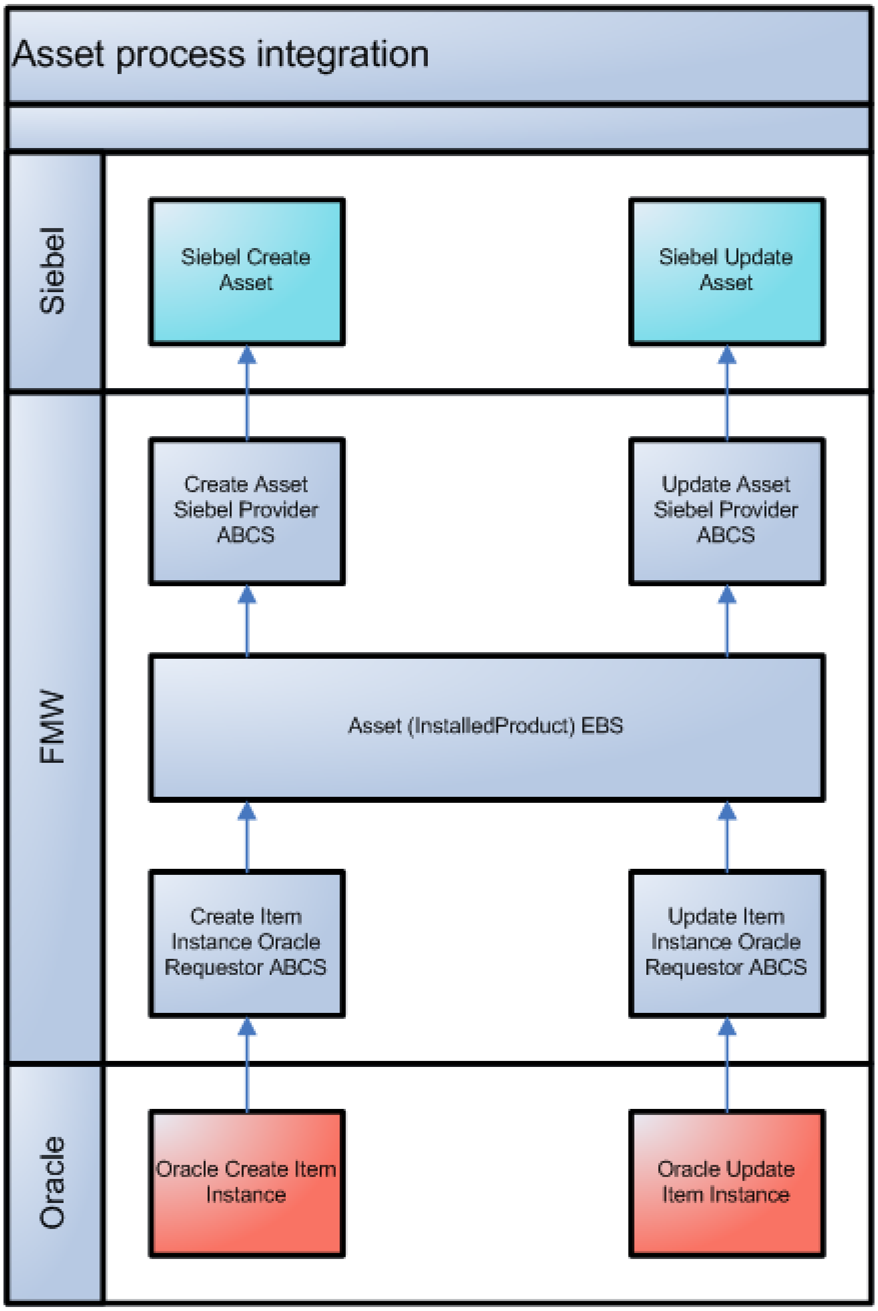
1. After assets are synchronized to Siebel CRM, they cannot be changed in CRM.
2. In Oracle EBS, Item Instances are created and updated for items whose Track in Installed Base attribute is set to True.
3. Only customer-owned (“Organization” Party) Item Instances (Assets) are synchronized from Oracle EBS to Siebel CRM.
4. In Siebel CRM, the asset is created on synchronization from Oracle EBS, and not by any workflow process in Siebel when the order status is completed.
5. Since asset synchronization is one-way from Oracle EBS to Siebel CRM, assets should not be updated in Siebel.

This diagram illustrates where the asset integration flow occurs in the Order to Cash: Siebel CRM – EBS process integration pack (PIP):



Asset Management integration flow in the Order to Cash: Siebel CRM – EBS PIP

This diagram illustrates the Create Asset and Update Asset integration flows:


Asset process integration
Siebel
FMW
Oracle
Siebel Create Asset
Siebel Update Asset
Create Asset Siebel Provider ABCS
Update Asset Siebel Provider ABCS
Asset (InstalledProduct) EBS
Create Item Instance Oracle Requestor ABCS
Update Item Instance Oracle Requestor ABCS
Oracle Create Item Instance
Oracle Update Item Instance

Create Asset and Update Asset integration flows

Creating Assets

The Create Assets integration flow synchronizes new Item Instances from Oracle EBS to Siebel CRM Assets in realtime and it enables a Customer Service Representative (CSR) to share the asset information with customers.

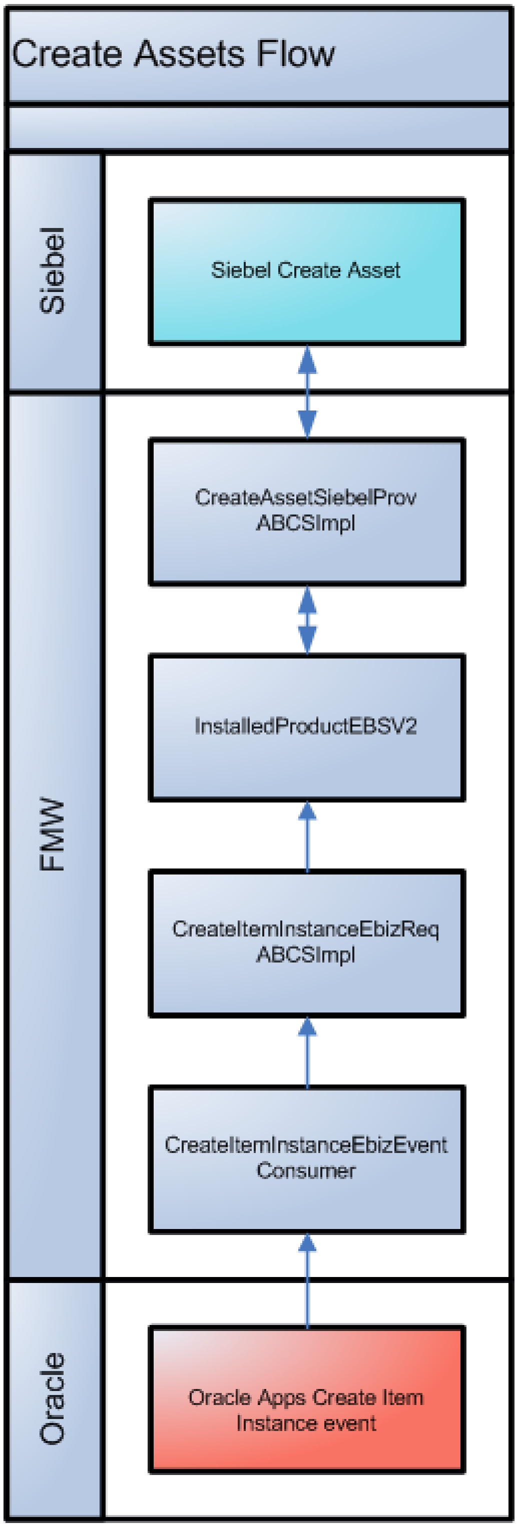
You create the Item Instance in the following ways in Oracle EBS:

- After sales orders are picked, released, and shipped, you create an install base item instance in Oracle EBS using auto-creation.
- You manually create an Item Instance 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

This diagram illustrates the Create Assets integration flow:


Create Assets Flow
Siebel
FMW
Oracle
Siebel Create Asset
CreateAssetSiebelProv ABCSImpl
InstalledProductEBSV2
CreateItemInstanceEbizReq ABCSImpl
CreateItemInstanceEbizEvent Consumer
Oracle Apps Create Item Instance event

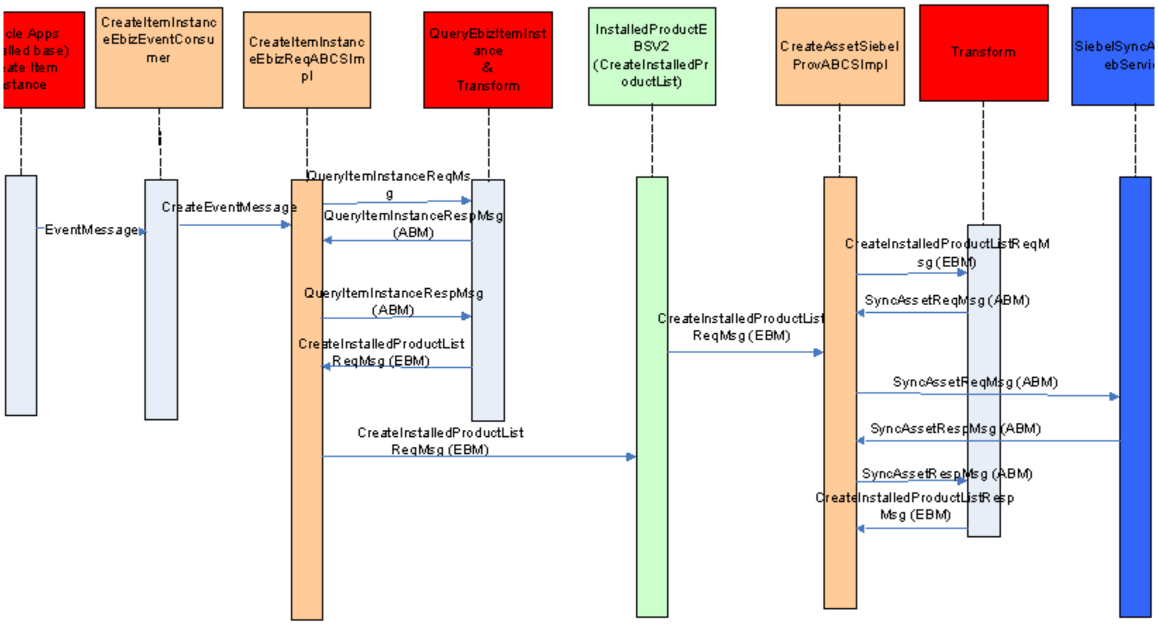
Create Assets Integration Flow

Create Assets Flow

This integration flow uses the following interfaces:

- CreateItemInstanceEbizEventConsumer
- QueryItemInstanceEbizAdapter
- CreateItemInstanceEbizReqABCSImpl
- InstalledProductEBSV2
- CreateAssetSiebelProvABCSImpl

This sequence diagram illustrates the Create Assets integration flow:


cle Apps
alled base)
ate Item
stance
CreateItemInstanceEbizEventConsumer
CreateItemInstanceEbizReqABCSImpl
QueryEbizItemInstance & Transform
InstalledProductEBSV2 (CreateInstalledProductList)
CreateAssetSiebelProvABCSImpl
Transform
SiebelSyncA ebServi
EventMessage
CreateEventMessage
QueryItemInstanceReqMsg
QueryItemInstanceRespMsg (ABM)
QueryItemInstanceRespMsg (ABM)
CreateInstalledProductList ReqMsg (EBM)
CreateInstalledProductList ReqMsg (EBM)
CreateInstalledProductList ReqMsg (EBM)
CreateInstalledProductListReqMsg (EBM)
SyncAssetReqMsg (ABM)
SyncAssetReqMsg (ABM)
SyncAssetRespMsg (ABM)
SyncAssetRespMsg (ABM)
CreateInstalledProductListRespMsg (EBM)

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 as well as 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 parent item instance will be built only as part of the update process.

4. The InstalledProductEBSV2 routes the CreateInstalledProductEBM to the Siebel provider ABCS implementation ‘CreateAssetSiebelProvABCSImpl.’
5. The CreateAssetSiebelProvABCSImpl transforms the ‘CreateInstalledProductEBM’ into the Siebel Asset ABM message ‘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 is linked with the common ID in the cross-reference) and then the message is sent back to the InstalledProductEBSV2.

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).

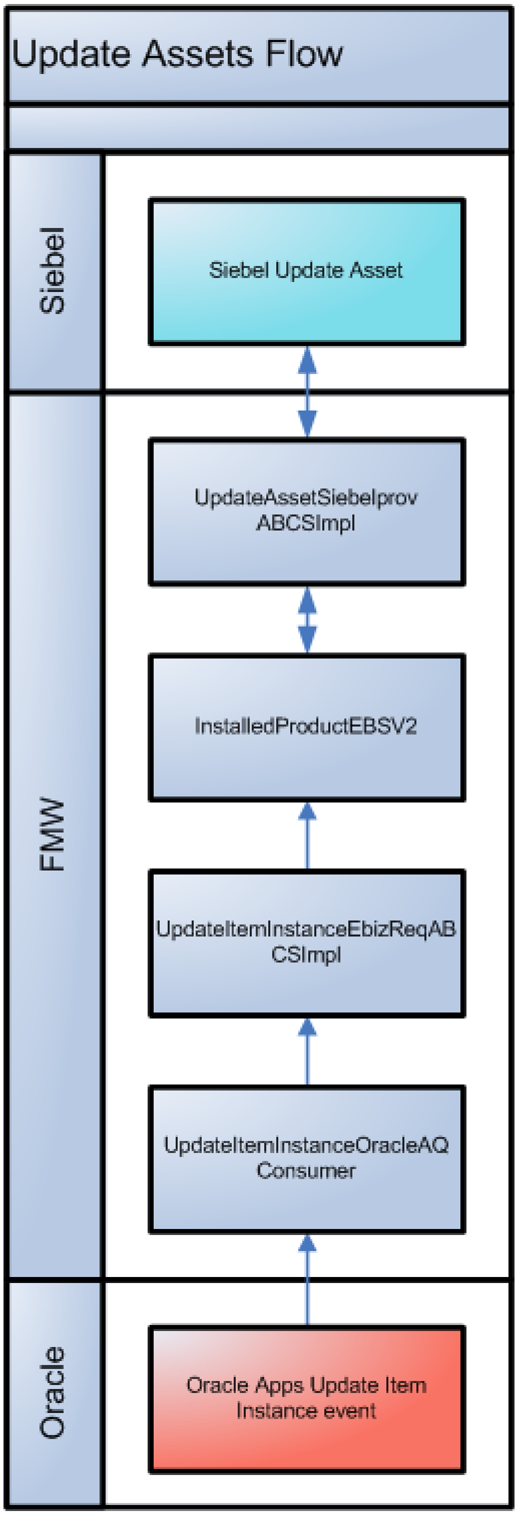
This one-way synchronization is from Oracle EBS to Siebel CRM. As Item Instances are updated in Oracle EBS, a business event is triggered that enables the synchronization of the latest asset status from Oracle EBS to Siebel CRM.

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.

This diagram illustrates the Update Assets integration flow:


Update Assets Flow
Siebel
FMW
Oracle
Siebel Update Asset
UpdateAssetSiebelprov ABCSImpl
InstalledProductEBSV2
UpdateItemInstanceEbizReqABCSImpl
UpdateItemInstanceOracleAQ Consumer
Oracle Apps Update Item Instance event

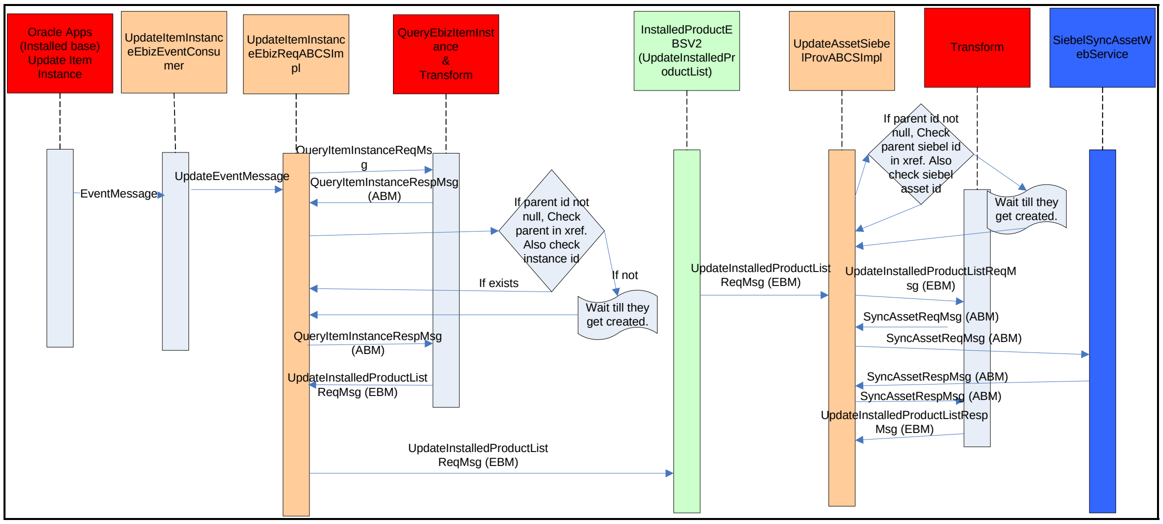
Update Assets integration flow

Update Assets Integration Flow

This integration flow uses the following interfaces:

- UpdateItemInstanceEbizEventConsumer
- QueryItemInstanceEbizAdapter
- UpdateItemInstanceEbizReqABCSImpl
- InstalledProductEBSV2
- UpdateAssetSiebelProvABCSImpl

This sequence diagram illustrates the Update Assets integration flow:


Oracle Apps (Installed base) Update Item Instance
UpdateItemInstanceEbizEventConsumer
UpdateItemInstanceEbizReqABCSImpl
QueryEbizItemInstance & Transform
InstalledProductEBSV2 (UpdateInstalledProductList)
UpdateAssetSiebelProvABCSImpl
Transform
SiebelSyncAssetWebService
EventMessage
UpdateEventMessage
QueryItemInstanceReqMsg
QueryItemInstanceRespMsg (ABM)
If parent id not null, Check parent in xref. Also check instance id
If exists
If not
Wait till they get created.
QueryItemInstanceRespMsg (ABM)
UpdateInstalledProductList ReqMsg (EBM)
UpdateInstalledProductList ReqMsg (EBM)
UpdateInstalledProductList ReqMsg (EBM)
If parent id not null, Check parent siebel id in xref. Also check siebel asset id
Wait till they get created.
UpdateInstalledProductListReqMsg (EBM)
SyncAssetReqMsg (ABM)
SyncAssetReqMsg (ABM)
SyncAssetRespMsg (ABM)
SyncAssetRespMsg (ABM)
UpdateInstalledProductListRespMsg (EBM)

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 building relationships to represent a BOM structure.

   The event is raised for simple as well as 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 to be created by the Create process, which must be initiated by the Create event from Oracle EBS. Once created, it 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.

Data Requirements

Product and Customer data must already be synchronized for the asset integration flow.

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 the Siebel CRM Integration Pack for Oracle Order Management Implementation Guide Addendum

Oracle EBS Interfaces

The Oracle EBS interface for the CreateInstalledProduct flow and the UpdateInstalledProduct flow is CSI_ASSET_INTEGRATION_V

For more information about EBS web services, see the following Oracle EBS references:
E-Business Suite Electronic Technical Reference Manual (eTRM) located on My Oracle Support under the E-Business Suite Information Center,
Oracle Integration Repository located at http://irep.oracle.com, Oracle Applications Release 11.5.10+ Online Documentation Library, located on the Oracle Technology Network (http://www.oracle.com/technology/documentation/applications.html).

Core AIA Components

The assets integration uses the following delivered core components:

- InstalledProductEBO
- CreateInstalledProductListEBM
- UpdateInstalledProductListEBM

The core EBO and EBM XSD files can be located by EBO within this parent folder: http://[HOST:PORT]/AIAComponents/EnterpriseObjectLibrary/Core/EBO/.

The core EBS WSDL files can be located by EBO within this parent folder: http://[HOST:PORT]/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/.

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

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

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

For more information, see *Oracle Application Integration Architecture - Foundation Pack: Integration Developer’s Guide*, “Extensibility for Oracle AIA Artifacts”

Integration Services

The following services are delivered with the Process Integration for Asset Management:

- InstalledProductEBSV2

- QueryItemInstanceEbizAdapter
- QueryItemInstanceEbizR12VersionAdapter
- CreateItemInstanceEbizEventConsumer
- CreateItemInstanceEbizReqABCSImpl
- CreateAssetSiebelProvABCSImpl
- UpdateItemInstanceEbizEventConsumer
- UpdateItemInstanceEbizReqABCSImpl
- UpdateAssetSiebelProvABCSImpl

InstalledProductEBSV2

The InstalledProductEBSV2 is an ESB routing service that exposes all of the enterprise operations that can be performed with the InstalledProductEBO. All of the integration flows for Order-to-Cash PIP 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

QueryItemInstanceEbizAdapter

The QueryItemInstanceEbizAdapter is an ESB 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 SOAP web service call.

QueryItemInstanceEbizR12VersionAdapter

The QueryItemInstanceEbizR12VersionAdapter is an adapter service. This queries the R12 Ebusiness suite view for Item Instance. This adapter call from the ABCSReqImpl becomes a SOAP web service call. This service is specific for R12 Ebusiness suite instance.

CreateItemInstanceEbizEventConsumer

The CreateItemInstanceEbizEventConsumer is an ESB 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.

CreateItemInstanceEbizReqABCSImpl

The CreateItemInstanceEbizReqABCSImpl is a 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 needed: Oracle EBS Item Instance ABM to CreateInstalledProductListEBM

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 needed:

- Installed ProductList EBM to CreateAsset ABM
- CreateAsset response ABM to CreateInstalledProductList response EBM

UpdateItemInstanceEbizEventConsumer

The UpdateItemInstanceEbizEventConsumer is an ESB 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.

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 needed: Ebiz Item Instance ABM to UpdateInstalledProductEBM

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.

Chapter 13: Implementing the Order to Cash: Siebel CRM – EBS Process Integration Pack

This chapter discusses how to:

- Configure Siebel CRM
- Get Inventory Location Details in Oracle EBS
- Set up organizations in Siebel Customer Relationship Management (CRM)
- Set up cross-references for Siebel IDs and Oracle E-Business Suite (Oracle EBS) entities
- Set up application context definitions for Oracle EBS
- Set up Oracle Configurator
- Work with cross-references
- Work with domain value maps (DVMs)
- Enable Oracle EBS events
- Create Oracle EBS system profiles
- Schedule concurrent processes
- Set a property in Oracle Transportation Management (OTM)
- Configure the Payment Authorization integration
- Handle errors
- View EBO Implementation Maps (EIMs)
- Set configuration properties

Siebel CRM and Oracle EBS must be set up in order for the Order to Cash: Siebel CRM – EBS process integration pack (PIP) to work properly. This setup includes setting up organizations, inventory locations, Oracle Configurator, and so on. This chapter describes these setups in detail.

Configuring Siebel CRM

After installing and configuring Siebel CRM and any required patched as described in Software Requirements for the Order to Cash: Siebel CRM – EBS PIP, perform the following configuration steps within your Siebel CRM system.

To configure Siebel CRM for the Order to Cash: Siebel CRM – EBS PIP:

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*". Results will list 14 web services.

   Of the results, the following services need to be considered:

   a. SyncCustomerSiebelEventAggregator

   b. RequestProductStructureSiebelJMSProducer

   c. ProcessSalesOrderSiebelJMSProducerV2

   d. CreditCheckSalesOrderSiebelReqABCSImplService

   e. CalculateShippingChargeSalesOrderSiebelReqABCSImplService

   f. PaymentAuthorizationSalesOrderSiebelReqABCSImplService

   g. ProcessQuoteSiebelJMSProducer

   h. CheckATPSalesOrderSiebelReqABCSImplService

   i. ConfiguratorUserLangSiebelAdapter

   j. 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://adc60118fems.us.oracle.com:7872/event/AIASystem/Ebiz/ABCS/Con
   figuratorCopyConfigEbizAdapter
   ```

6. Activate the following workflows:

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

Workflow Name	Status	Project
Oracle Verify Header (Quote)	New	Web Service Integration
Oracle Verify Item (Order)	New	Web Service Integration
Oracle Verify Item (Quote)	New	Web Service Integration
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 following component groups:

Component Group Name	Alias
Enterprise Application Integration	EAI
Siebel CME (HTIM)	Communications
Siebel ISS	ISS
Workflow Management	Workflow

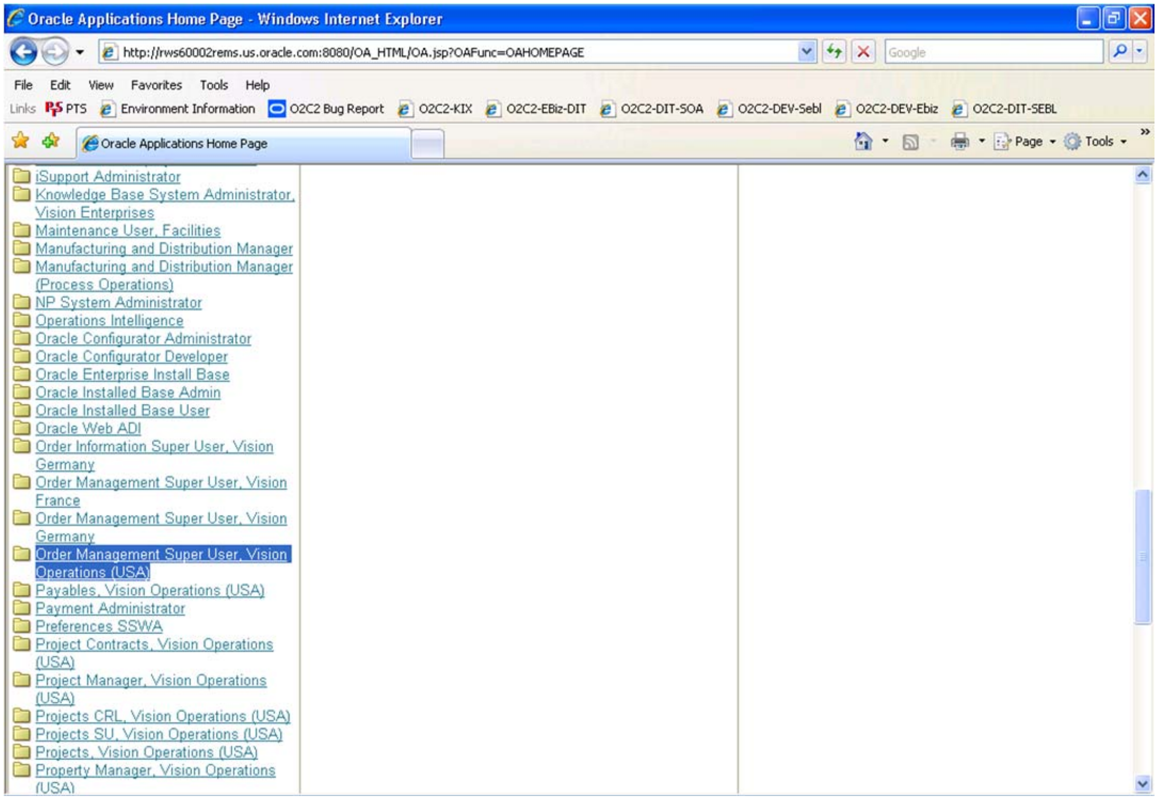
For more information about enabling component groups, see *Siebel System Administration Guide,* available on Oracle Technology Network

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.


Oracle Applications Home Page - Windows Internet Explorer
http://rws60002rems.us.oracle.com:8080/OA_HTML/OA.jsp?OAFunc=OAHOMEPAGE
Google
File Edit View Favorites Tools Help
Links PTS Environment Information O2C2 Bug Report O2C2-KIX O2C2-EBiz-DIT O2C2-DIT-SOA O2C2-DEV-Sebl O2C2-DEV-Ebiz O2C2-DIT-SEBL
Oracle Applications Home Page
Page Tools
iSupport Administrator
Knowledge Base System Administrator, Vision Enterprises
Maintenance User, Facilities
Manufacturing and Distribution Manager
Manufacturing and Distribution Manager (Process Operations)
NP System Administrator
Operations Intelligence
Oracle Configurator Administrator
Oracle Configurator Developer
Oracle Enterprise Install Base
Oracle Installed Base Admin
Oracle Installed Base User
Oracle Web ADI
Order Information Super User, Vision Germany
Order Management Super User, Vision France
Order Management Super User, Vision Germany
Order Management Super User, Vision Operations (USA)
Payables, Vision Operations (USA)
Payment Administrator
Preferences SSWA
Project Contracts, Vision Operations (USA)
Project Manager, Vision Operations (USA)
Projects CRL, Vision Operations (USA)
Projects SU, Vision Operations (USA)
Projects, Vision Operations (USA)
Property Manager, Vision Operations (USA)

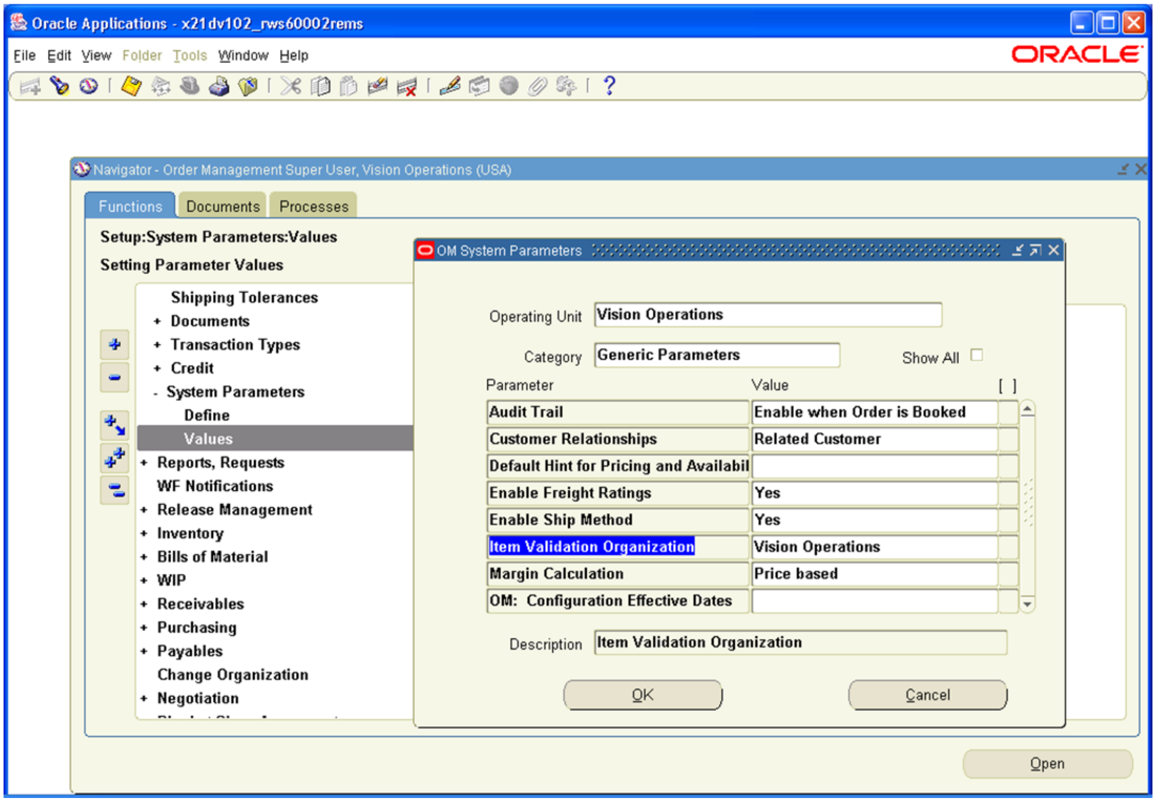
Example of the Oracle Applications home page

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 illustrated in the following page example.

   c. Write down the name of the Item Validation Organization associated with the parameter Item Validation Organization, for example, Vision Operations, Vision Germany, and Vision France.



Operating Unit: Vision Operations

Category: Generic Parameters   Show All □

Parameter	Value
Audit Trail	Enable when Order is Booked
Customer Relationships	Related Customer
Default Hint for Pricing and Availabil	
Enable Freight Ratings	Yes
Enable Ship Method	Yes
Item Validation Organization	Vision Operations
Margin Calculation	Price based
OM: Configuration Effective Dates	

Description: Item Validation Organization

Example of system parameters query

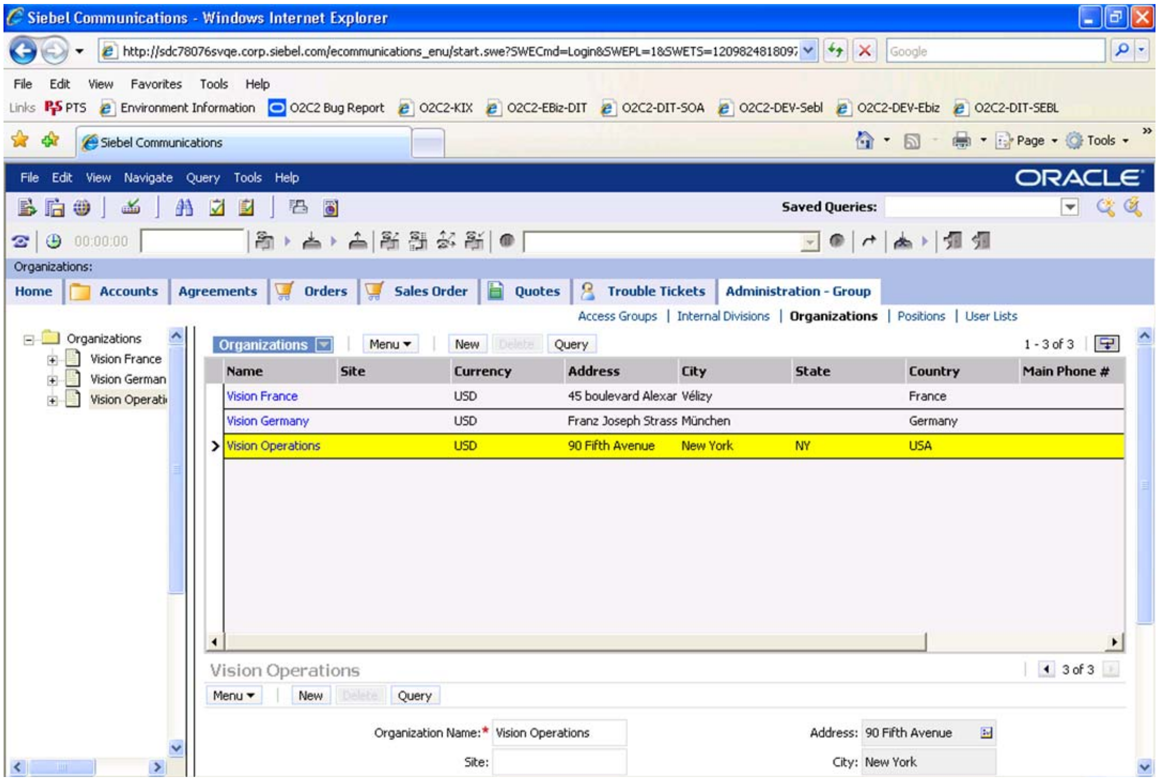
Setting Up Organizations in Siebel CRM

The following sections describe how to set up inventory locations and organizations in Siebel CRM.

Setting Up Organizations in Siebel CRM

To map Siebel organizations to 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 illustrated in the following example.


Siebel Communications - Windows Internet Explorer
http://sdc78076svqe.corp.siebel.com/ecommunications_enu/start.swe?SWECmd=Login&SWEPL=1&SWETS=120982481809
File Edit View Favorites Tools Help
Links PTS Environment Information O2C2 Bug Report O2C2-KIX O2C2-EBiz-DIT O2C2-DIT-SOA O2C2-DEV-Sebl O2C2-DEV-Ebiz O2C2-DIT-SEBL
Siebel Communications
Page Tools
File Edit View Navigate Query Tools Help
ORACLE
Saved Queries:
Organizations:
Home Accounts Agreements Orders Sales Order Quotes Trouble Tickets Administration - Group
Access Groups | Internal Divisions | Organizations | Positions | User Lists
Organizations
Vision France
Vision German
Vision Operati
Organizations Menu New Delete Query
1 - 3 of 3
Vision Operations
Menu New Delete Query
3 of 3

Name	Site	Currency	Address	City	State	Country	Main Phone #
Vision France		USD	45 boulevard Alexar	Vélizy		France	
Vision Germany		USD	Franz Joseph Strass	München		Germany	
Vision Operations		USD	90 Fifth Avenue	New York	NY	USA	

Organization Name:* Vision Operations

Site:

Address: 90 Fifth Avenue

City: New York

Example of setting up organizations in Siebel CRM

Note: Create an organization record with Name only. Leave the Site field blank.

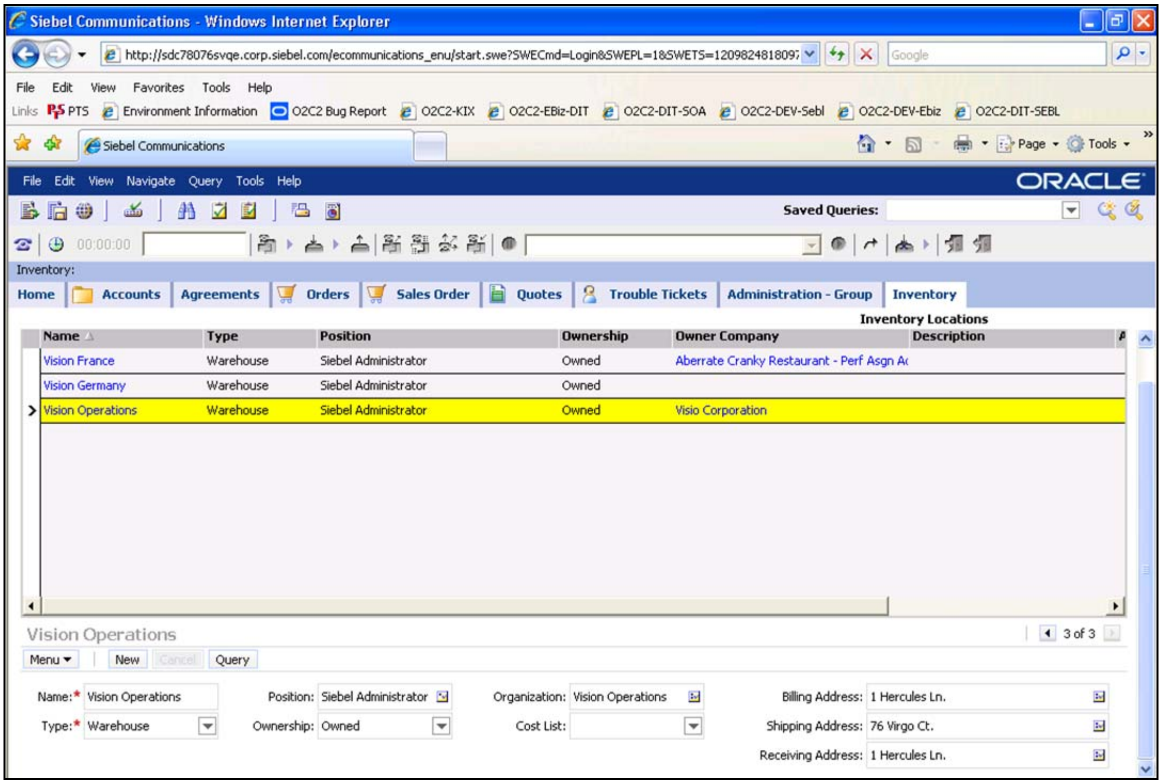
Setting Up Inventory Locations in Siebel CRM

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 exactly the same name that was identified in Oracle. An example follows.


Siebel Communications - Windows Internet Explorer
http://sdc78076svqe.corp.siebel.com/ecommunications_enu/start.swe?SWECmd=Login&SWEPL=1&SWETS=120982481809
File Edit View Favorites Tools Help
Links PTS Environment Information O2C2 Bug Report O2C2-KIX O2C2-EBiz-DIT O2C2-DIT-SOA O2C2-DEV-Sebl O2C2-DEV-Ebiz O2C2-DIT-SEBL
Siebel Communications
Page Tools
File Edit View Navigate Query Tools Help
ORACLE
Saved Queries:
00:00:00
Inventory:
Home Accounts Agreements Orders Sales Order Quotes Trouble Tickets Administration - Group Inventory
Inventory Locations
3 of 3
Menu New Cancel Query

Name	Type	Position	Ownership	Owner Company	Description
Vision France	Warehouse	Siebel Administrator	Owned	Aberrate Cranky Restaurant - Perf Asgn Ac	
Vision Germany	Warehouse	Siebel Administrator	Owned		
Vision Operations	Warehouse	Siebel Administrator	Owned	Visio Corporation	

Vision Operations

Name:* Vision Operations
Type:* Warehouse
Position: Siebel Administrator
Ownership: Owned
Organization: Vision Operations
Cost List:
Billing Address: 1 Hercules Ln.
Shipping Address: 76 Virgo Ct.
Receiving Address: 1 Hercules Ln.

Example of Inventory Locations

5. Make sure the Organization and Inventory Location name in Siebel matches the names in Oracle EBS as the bulk load references them by name and not by ID.

Setting Up Cross-References for Siebel IDs and Oracle EBS Entities

You create cross-references after you create inventory locations both in Siebel CRM and in Oracle EBS. In Siebel, you create inventory locations manually.

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%')
   ```

Identifying EBS Entities

To get the operating unit details:

1. Log in to 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 exact 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
   ```

Populating Cross-References

To populate the cross-references:

1. Create Organization.xml and Inventory_location_id.xml using the following samples. Update the XML files with the IDs from Siebel CRM.

   Oracle IDs for this will remain the same until different organizations/Inventory Organizations are selected, created, or both.

   Sample Organization.xml:

   ```
   <xref xmlns="http://xmlns.oracle.com/xref">
      <table name="ORGANIZATION_ID">
            <columns>
                  <column name="EBIZ_01"/>
                  <column name="SEBL_01"/>
            </columns>
            <rows>
                  <row>
                        <cell colName="EBIZ_01">204</cell>
                        <cell colName="SEBL_01">88-25CHZ</cell>
                  </row>
                  <row>
                        <cell colName="EBIZ_01">888</cell>
                        <cell colName="SEBL_01">88-25CIA</cell>
                  </row>
                  <row>
                        <cell colName="EBIZ_01">911</cell>
                        <cell colName="SEBL_01">88-25CI5</cell>
                  </row>
            </rows>
      </table>
   </xref>
   ```

Sample Inventory_location_id.xml:

```
<xref xmlns="http://xmlns.oracle.com/xref">
   <table name="INVENTORY_LOCATION_ID">
         <columns>
               <column name="EBIZ_01"/>
               <column name="SEBL_01"/>
         </columns>
         <rows>
               <row>
                     <cell colName="EBIZ_01">204</cell>
                     <cell colName="SEBL_01">88-5I54T</cell>
               </row>
               <row>
                     <cell colName="EBIZ_01">888</cell>
                     <cell colName="SEBL_01">88-29CMT</cell>
               </row>
               <row>
                     <cell colName="EBIZ_01">911</cell>
                     <cell colName="SEBL_01">88-30GX4</cell>
               </row>
         </rows>
   </table>
</xref>
```

For more information about creating the cross references, see *Oracle Enterprise Service Bus Developer's Guide 10g (10.1.3.4.0),* Creating Cross References.

2. Run the xrefimport tool to import the cross-references.
3. Copy the files to a temporary directory on the SOA server.
4. Telnet to the SOA Server and change dir to the xrefimport tool home:

```
-bash-3.00$ cd $SOA_HOME/integration/esb/bin
```

5. Set the following env variables:

```
-bash-3.00$ export OC4J_USERNAME=oc4jadmin
-bash-3.00$ export OC4J_PASSWORD=welcome1
-bash-3.00$ export DB_USER=aia
-bash-3.00$ export DB_PASSWORD=aia
-bash-3.00$ export
DB_URL="jdbc:oracle:thin:@adc60119fems.us.oracle.com:1549:o2c2sysa"
```

6. Run the import for ORGANIZATION xref using the following command:

```
bash-3.00$. xrefimport.sh -file ~/orginvsetup/Organization.xml -
generate COMMON
```

7. Run the import for INV LOC xref using the following command:

```
. xrefimport.sh -file ~/orginvsetup/Inventory_location_id.xml -
generate COMMON
```

Validating Cross-References

To validate the cross-references:

1. Log in to the AIA XREF database.
2. Query the Table XREF_DATA to confirm that every organization and every inventory location used in the XML files has three records.

   Use the following query:

   ```
   select value||'::'||Xref_column_name from xref_Data where
   row_number in (select row_number from xref_data where
   xref_table_name = 'INVENTORY_LOCATION_ID'
   and value in ('204'))
   ```
3. Replace the value for the inventory locations and replace INVENTORY_LOCATION_ID with ORGANIZATION_ID for the operating units you selected. (The number of operating units and inventory locations depends on your setup.)

For more information about cross-references, see Working with Cross-References

Setting Up Application Context Definitions for Oracle EBS

This section describes how to set up the derivation of the context in flows where Oracle EBS API or service calls are made.

Application context (or API) for Oracle EBS calls is 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) as described in the “Setting Up Cross-References for Siebel IDs and Oracle EBS Entities” section, and cross-references in the ORGANIZATION_ID XREF table. In any flow where 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 config property:

Property Name: TransformAppContextEbizService.DefaultOperatingUnit

Service Name: TransformAppcontextEbizService

If required, users can also create cross-reference between applications using the USER_NAME XREF table. In any flow where the user context of source application is available and is mapped in the USER_NAME XREF table, looking up 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 config 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 priviliges 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)

Setting Up Oracle Configurator

For the Quotes and the Orders integrations, you must change the runtime configuration in Siebel CRM to invoke the Oracle Configurator from Siebel CRM.

For more information on how to set up Siebel QF for ACR 404, see *Siebel CRM Integration Pack for Oracle Order Management Implementation Addendum*. This document can be located here: https://metalink3.oracle.com/od/faces/secure/km/DocumentDisplay.jspx?id=728946.1

Changing the Runtime Configuration to Invoke Oracle Configurator from Siebel CRM

You must make the following runtime configuration changes before you can launch Oracle Configurator.

To change the runtime configuration to invoke Oracle Configurator from Siebel CRM:

1. Log in to the 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. Make sure they are in the active state by querying in the bottom applet.
3. Go to site 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 drop-down menu.

b. Add a new host entry:

Host Name: <Oracle EBIZ host name:port number> (For example, qapache.us.oracle.com:3710)

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 server-side business component code relies on this name.

URL: http://OracleConfigurator/OA_HTML/CfgSebl.jsp

Select the host name, for example, qapache.us.oracle.com:3710.

Fixup Name: Default

SSO Disposition: IFRAME

In the bottom applet, add arguments to the URL:
Create a new record with these values:
Name: InitMessage
Req Arg: Selected
Argument Type: Profile Attribute
Argument Value: CZInitMessage
Append as argument: Selected

Add another new row:
Name: PostRequest
Required Arg: Selected
Argument Type: Command
Argument Value: PostRequest
Append as argument: Deselected

d. Implement the following two steps to set up the single sign on (SSO) login to Oracle EBS.

The SSO login is the preferred mode of accessing Oracle EBS. If SSO login is not set up, then the Siebel end user will have to know and enter Oracle login credentials at a login page. If SSO is going to be used, you must complete Step 5.

e. 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 will require the user to log in to an Oracle Applications for Configurator session. In Step 5, map the Siebel user to the Oracle login credentials.

f. Continue to add arguments to the same URL:

Name: Username
Required Arg: Selected
Argument Type: Command
Argument Value: UserLoginId
Append as argument: Selected

g. Add the final argument

Name: Password
Required Arg: Selected
Argument Type: Command
Argument Value: UserLoginPassword
Append as argument: Selected

h. 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

i. In the bottom applet, SSO System Users, add records for the Siebel users who will 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. Note that these workflows already 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*

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:

   a. AutoAgreement(Data Map Component:Line Item)

b. AutoAgreeFromOrder(Data Map Component Name:Line Items)

c. CopyOrder(Data Map Component Name:Line Item)

d. CopyQuote(Data Map Component Name:Line Item)

e. OrderToTemplate(Data Map Component Name:Line Item)

f. QuoteToSalesOrder(Data Map Component Name:Line Item)

g. QuoteToServiceOrder(Data Map Component Name:Line Item)

h. QuoteToTemplate(Data Map Component Name:Line Item)

i. ReviseAgreement(Data Map Component Name:Line Item)

j. ReviseOrder(Data Map Component Name:Line Item)

k. ReviseQuote(Data Map Component Name:Line Item)

l. TemplateToOrder(Data Map Component Name:Line Item)

m. TemplateToQuote(Data Map Component Name:Line Item)

Setting Up the DoCompression Parameter

To set up the DoCompression parameter:

1. Stop the web server (viz. 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.

Adding Siebel Custom Applications to Oracle Applications

You must create several custom Oracle Applications to allow models to be accessed from Siebel. These custom applications are used during the publication phase of Oracle Configurator Model development cycle.

To add Siebel custom applications to Oracle applications:

1. Log in to the Oracle applications with the system administrator credentials.

2. Select Application, Register.

Forms will 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

   The new applications must be added 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. Once the program runs, these new applications will be available during model publication.

For more information about the process of developing models for calling applications, see the Oracle Configurator product documentation

Enabling Siebel Eligibility and Compatibility in Oracle Configurator

By default, Siebel Eligibility and Compatibility are not enabled when launching 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 configuration model's UI 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 one of the following cz.activemodel values:

   a. ecv - Display violations as a validation failure

   b. 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 will be 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

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:

   a. NAME: ABS-DESKTOP

   b. MEANING: ABS Custom Look And Feel

   c. 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.

Displaying Images and Icons from Oracle Configurator in Siebel

To solve problems viewing images, icons, and so forth from Oracle Configurator in Siebel:

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 EBS server is correct:

   a. Log on to Oracle EBS.

   b. Add/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 hGrid component.

   The correct images should appear on this page.

5. Clear the browser cache before performing any test on your machine.

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 Application Integration Architecture - Foundation Pack - Integration Developer's Guide and the Oracle Cross Reference User Guide.

This table lists the Order to Cash: Siebel CRM – EBS 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
	SEBL_01		
CUSTOMERPARTY_ACCOUNTID	EBIZ_01	Customer account IDs	Lookup during sales order flow, lookup/populated during customer flow
	COMMON		Populated by sales order/customer flow
	SEBL_01		Lookup during asset flow

Table	Column	Description	Usage
CUSTOMERPARTY_ADDRESSID	EBIZ_01	Address (location) IDs	Lookup during sales order flow, lookup/populated during customer flow
	COMMON		Populated by sales order/customer flow
	SEBL_01		
CUSTOMERPARTY_CONTACTID	EBIZ_01	Contact/Person Ids	Lookup during sales order flow, lookup/populated during customer flow
	COMMON		Populated by sales order/customer flow
	SEBL_01		
CUSTOMERPARTY_LOCATIONREFID	EBIZ_01	Address (location) IDs	Lookup during sales order flow, lookup/populated during customer flow
	COMMON		Populated by sales order/customer flow
	SEBL_01		
CUSTOMERPARTY_PARTYLOCATIONID	EBIZ_01	Party address Ids	Lookup during sales order flow, lookup/populated during customer flow
	COMMON		Populated by sales order/customer flow
	SEBL_01		
CUSTOMERPARTY_ACCOUNT_PHONECOMMID	EBIZ_01	Account Phone contact points	lookup/populated during customer flow
	COMMON		
	SEBL_01		
CUSTOMERPARTY_ACCOUNT_FAXCOMMID	EBIZ_01 column	Account Fax contact points	lookup/populated during customer flow
	COMMON		
	SEBL_01		
CUSTOMERPARTY_ACCOUNT_WEBCOMMID	EBIZ_01 column	Account Email/Web contact points	lookup/populated during customer flow
	COMMON		
	SEBL_01		

Table	Column	Description	Usage
CUSTOMERPARTY_CONTACT_PHONECOMMID	EBIZ_01 column	Contact's Phone contact points	lookup/populated during customer flow
	COMMON		
	SEBL_01		
CUSTOMERPARTY_CONTACT_FAXCOMMID	EBIZ_01	Contact's Fax contact points	lookup/populated during customer flow
	COMMON		
	SEBL_01		
CUSTOMERPARTY_CONTACT_EMAILCOMMID	EBIZ_01	Contact's Email/Web contact points	lookup/populated during customer flow
	COMMON		
	SEBL_01		
CUSTOMERPARTY_PARTYCONTACTID	EBIZ_01	Party contact IDs	Lookup during sales order flow, lookup/populated during customer flow
	COMMON		
	SEBL_01		
SALESORDER_ID	EBIZ_01	Sales Order ID	Populated by sales order flow
	COMMON		
	SEBL_01		
SALESORDER_LINEID	EBIZ_01	Sales Order Line ID	Populated by sales order flow
	COMMON		
	SEBL_01		
QUOTE_ID	EBIZ_01	Quote ID	Populated by sales order flow
	COMMON		
	SEBL_01		
QUOTE_LINEID	EBIZ_01	Quote Line ID	Populated by sales order flow

Table	Column	Description	Usage
	COMMON		
	SEBL_01		
ORGANIZATION_ID	EBIZ_01	Organization/Business Unit IDs	Lookup during sales order/customer flow
	COMMON		Lookup during asset flow
	SEBL_01		
PRICELIST_ID	EBIZ_01	Pricelist ID	Lookup during sales order flow
	COMMON		Populated by Price List bulk load
	SEBL_01		
ITEM_ITEMID	EBIZ_01	Item/Product ID	Lookup during sales order flow
	COMMON		Lookup during asset flow
	SEBL_01		
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

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 needed. Transactional business processes never update DVMs; they only read from them. DVMs are stored in XML files and cached in memory at runtime.

DVM types are seeded for the Order to Cash: Siebel CRM – EBS 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.

This table lists the DVMs for the Order to Cash: Siebel CRM – EBS PIP:

DVM Type	DVM Column Name	Comments
ADDRESS_COUNTRYID	EBIZ_01, COMMON, SEBL_01	Country codes
STATE	EBIZ_01, COMMON, SEBL_01	State codes
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_METHOD	EBIZ_01, COMMON, SEBL_01	Shipping method codes
CURRENCY_CODE	EBIZ_01, COMMON, SEBL_01	Currency codes
ORDER_CARRIER_TYPE_CODE	EBIZ_01, COMMON, SEBL_01,GLOG_01	Carrier codes (such as FedEx,

DVM Type	DVM Column Name	Comments
		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 order is revised or cancelled.

Mapping ATP Check Status Codes to Messages

In EBusiness Suite, the result of the ATP Check is communicated back as a status code. In order to map these status codes to appropriate error messages, mappings need to be established in the ATPCHECK_STATUS DVM. Out of the box, mappings for the most common status codes are pre-populated.

The mapping that exists in EBusiness Suite, can be looked up in the EBusiness Suite Lookup type: 'MTL_DEMAND_INTERFACE_ERRORS'.

Enabling Oracle EBS Events

Log into the Oracle EBS ‘Workflow Administrator’ responsibility and make sure 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

Creating Oracle EBS System Profiles

You create specific profile options for the Customer and Asset Management process integrations.

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:

For E-Business Suite 11.5.10.2:

- HZ: Execute API Callouts to All Events Enabled or Only Business Object Events Enabled
- HZ: Format Business Object Business Events as Bulk to N

For E-Business Suite 12.1.1:

- HZ: Raise API Events to All Events Enabled or Only Business Object Events Enabled
- HZ: Format Business Object Business Events as Bulk to ‘N’

- HZ: Generate Party Number to 'Yes'
- HZ: Generate Party Site Number to 'Yes'

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:
   - **a.** CSI: Raise Business Event For Customer Owned Instances to Yes
   - **b.** CSI: Auto-split Instances During Instantiation to No

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:
   - **a.** Order line is Booked and order line is not a 'RLM Line'
   - **b.** Order line is 'PO Approved'
4. Clear the Constraint check box.
5. Save and close the form.

Creating Profile Values for the Order Management Integration

To create profile values for the Order Management integration post-installation:

1. Log in to the 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 into 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.

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.

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: Describing the Shipping Charges Integration Flow

Configuring the Payment Authorization Integration

To configure the Payment Authorization integration:

1. Set IBY: ECAPP URL (IBY profile name) to:

   http:// <host machine name.us.oracle.com>:<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.us.oracle.com>:<port number>/oa_servlets/ibyecapp" (Example: http qapache.us.oracle.com:3908/oa_servlets/ibyecapp)

2. Set the servlet base URL to:

   http:// <host machine name.us.oracle.com>:<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.us.oracle.com>:<port number>/oa_servlets
   (Example: http://qapache.us.oracle.com:3908/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"

Handling Errors

Based on the roles defined for the services, the system sends email notifications if a service errors out. No AIA-specific errors are thrown by the process integration for Product Management services.

Based on the roles defined for the services, the system sends email notifications if a service errors out. This table lists the errors thrown by the Order to Cash: Siebel CRM – EBS 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 already been synchronized.
AIA_ERR_AIAO2C2_1006	Account could not be queried. Please make sure 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.

Error Code	Message Text
AIA_ERR_AIAO2C2_1008	Credit Check failed. Please contact your System Administrator.

For more information about the errors thrown by Siebel CRM or Oracle EBS, see that product's documentation.

For more information about AIA error handling, see the *Oracle Application Integration Architecture - Foundation Pack: Core Infrastructure Components Guide,* "Setting Up and Using Error Handling and Logging"

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: Siebel CRM – EBS PIP.

Actor Roles and Users

- Role: OracleSiebelAdmin User: OracleSiebelAdminUser
- Role: OracleBRMAdmin User: OracleBRMAdminUser
- Role: AIAIntegrationAdmin User: AIAIntegrationAdminUser

FYI Roles and Users

- Role: OracleSiebelCSR User: OracleSiebelCSRUser
- Role: OracleBRMCSR User: OracleBRMCSRUser

The default password set for all users is welcome1.

For more information about setting up error notifications using these values, see *Oracle Application Integration Architecture -– Foundation Pack: Core Infrastructure Components Guide,* "Setting Up Error Notifications and Trace Logging"

Viewing EBO Implementation Maps (EIMs)

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

Setting Configuration Properties

Set these properties in the AIAConfigurationProperties.xml file. The file is located in <aia.home>/config/.

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

For more information, see the *Oracle Application Integration Architecture Core Components Guide*, "Working with the BSR," Loading Oracle AIA Configuration File Updates

Settings for the 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.
BULKLOAD.DEFAULT.SOURCE	No default value	Ebiz system code (defined in BSR) which is the source application for Initial loads
BULKLOAD.DEFAULT.TARGET	No default value	Siebel system code (defined in BSR) which is the target application for initial loads
EBIZ_01.SERVER_TIMEZONE	No default value	Time zone code

Settings for the InterfaceCustomerToFulfillmentEBF service property:

Property Name	Value/Default Value	Description
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 time out and aborts.
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 to the target service.

Property Name	Value/Default Value	Description
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 to the target service.
Routing.SalesOrderOrchestrationResponseEBSV2.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 to the target service.
Routing.CustomerPartyEBSV2.SyncCustomerPartyList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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.SalesOrderOrchestrationResponseEBSV2.InterfaceCustomerToFulfillmentResponse.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncrequestrecipient	CAVS Endpoint URI, when CAVS is enabled.

Settings for the SyncAccountSiebelReqABCSImpl service property:

Property Name	Value/Default Value	Description
Default.SystemID	No default value	Siebel system code (like SEBL_01,

Property Name	Value/Default Value	Description
		defined in BSR) 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 to the target service.
Routing.CustomerPartyEBSV2.SyncCustomerPartyList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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.
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) will be 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) will be invoked.
Account.ProcessUpdateEventsOnly	True	This property governs whether create events raised in the Siebel application are consumed or not. By default, only update events are consumed.
Routing.SWICustomerParty.SEBL_01.EndpointURI	http://<SIEBEL_HOST>/eai_enu/start.swe?SWEExtSource=WebService&amp;SWEExtCmd=Execute&amp;UserName=<USER_NAME>&amp;Password=<PASSWORD>	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 or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SWICustomerParty.CAVS.Endp	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServle	CAVS Endpoint URI, when CAVS is

Property Name	Value/Default Value	Description
ointURI	t/syncresponsesimulator	enabled.

Settings for the SyncCustomerPartyListEbizReqABCSImpl service property:

Property Name	Value/Default Value	Description
Default.SystemID	No default value	Ebiz system code (like EBIZ_01, defined in BSR) 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 or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property
Routing.QueryCustomerPartyListEbizCreate.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/QueryCustomerPartyListEbizCreateAdapter	Endpoint URI of the Ebiz adapter
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 or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.QueryCustomerPartyListEbizUpdate.EBIZ_01.EndpointURI r	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/QueryCustomerPartyListEbizUpdateAdapter	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 or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.CustomerPartyEBSV2.SyncCustomerPartyList.MessageProcessingInstru	Default value = PRODUCTION	Governs whether the message is routed to CAVS or to the specified

Property Name	Value/Default Value	Description
ction.EnvironmentCode		target service. Default value is PRODUCTION, which routes 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) will be 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) will be 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) will be invoked.

Settings for the QueryCustomerPartyListSiebelProvABCSImplV2 service property:

Property Name	Value/Default Value	Property
Default.SystemID	No default value	Siebel system code (like SEBL_01, defined in BSR) 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&amp;SWEExtCmd=Execute&amp;UserName=<USER_NAME>&amp;Password=<PASSWORD>	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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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.

Property Name	Value/Default Value	Property
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) will be 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) will be 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) will be 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) will be invoked.

Settings for the SyncCustomerPartyListEbizProvABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	No default value	Ebiz system code (like EBIZ_01, defined in BSR) to which data is synced into. 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>/event/AIASystem/Ebiz/ABCS/SyncCustomerPartyListEbizAdapter	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 or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.

Property Name	Value/Default Value	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 to the target service.
Routing.CustomerPartyResponseEBSV2.SyncCustomerPartyListResponse.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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) will be invoked.
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) will be 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) will be 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) will be invoked.

Settings for the SyncCustomerPartyListEbizProvABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	No default value	Ebiz system code (like EBIZ_01, defined in BSR) to which data is synced into. The System code value present in the EBM Header of the incoming message takes precedence over this property value.

Property Name	Value/Default Value	Property
Routing.SyncCustomerPartyListEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/SyncCustomerPartyListEbizAdapter	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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 to the target service.
Routing.CustomerPartyResponseEBSV2.SyncCustomerPartyListResponse.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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) will be invoked.
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) will be 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) will be invoked.
ABCSExtension.PostXformABMtoEBME	true/false Default = false	Governs whether ABCS Extension is

Property Name	Value/Default Value	Property
BM		enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) will be invoked.

Settings for the SyncCustomerPartyListSiebelProvABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	No Default value	Siebel system code (like SEBL_01, defined in BSR) to which data is synced into. 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&amp;SWEExtCmd=Execute&amp;UserName=<USER_NAME>&amp;Password=<PASSWORD>	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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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&amp;SWEExtCmd=Execute&amp;UserName=<USER_NAME>&amp;Password=<PASSWORD>	Siebel SWICustomerPartyService(Merge) service endpoint location. This is a SOAP endpoint URL. If the request message contains the target URL, then that takes precedence.
Routing.MergeCustomerPartyListSiebelService.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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.

Property Name	Value/Default Value	Property
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 to the target service.
Routing.CustomerPartyEBSV2.SyncCustomerPartyListResponse.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 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) will be 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) will be 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) will be 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) will be invoked.
ABCSExtension.PostInvokeSWIC	true/false Default = false	Governs whether ABCS

Property Name	Value/Default Value	Property
ustUpsertServiceABM		Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) will be invoked.

Settings for the MergeAccountEbizReqABCSImpl servic property:

Property Name	Value/Default Value	Property
Default.SystemID	No default value	Ebiz system code (like EBIZ_01, defined in BSR) from which requests originate for this process
Routing.QueryMergeAccountEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/QueryMergeAccountEbizAdapter	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 or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.QueryCustomerPartyEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/QueryCustomerPartyEbizAdapter	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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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.SyncCusto	true/false Default = false	Governs whether the service should

Property Name	Value/Default Value	Property
merPartyList.RouteToCAVS		route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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.
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) will be 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) will be 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) will be invoked.

Settings for the MergePartyEbizReqABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	No default value	Ebiz system code (like EBIZ_01, defined in BSR) from which requests originate for this process
Routing.QueryPartyMergeEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/QueryPartyMergeEbizAdapter	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 or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the

Property Name	Value/Default Value	Property
		CAVS.EndpointURI property.
Routing.QueryMergeOrgCustEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/QueryMergeOrgCustEbizAdapter	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 or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.QueryRelatedOrgCustEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/QueryRelatedOrgCustEbizAdapter	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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 to the target service.
Routing.CustomerPartyEBS.SyncCustomerPartyList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
ABCSExtension.PreXformABMToEBMA	true/false Default = false	Governs whether ABCS Extension is

Property Name	Value/Default Value	Property
BM		enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) will be 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) will be 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) will be invoked.

Settings for the InterfaceSalesOrderToFulfillmentEBF service property:

Property Name	Value/Default Value	Property
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	This property defines the value for RMA Order Types. A list of code values that represent RMA order type. During processing of orders with this type, Customer sync will 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 aborts. Duration to wait for the asynchronous callback from the InterfaceSalesOrderToCustomer service. If response is not received within the wait duration, a fault will occur.
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 aborts. Duration to wait for the asynchronous CreateSalesOrderResponse callback from the SalesOrderResponseEBS service. If response is not received within the wait duration, a fault will occur.
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 aborts. Duration to wait for the asynchronous UpdateSalesOrderResponse callback from the SalesOrderResponseEBS service. If response is not received within the wait duration, a fault will occur.

Property Name	Value/Default Value	Property
Routing.SalesOrderEBSV2.CreateSalesOrder.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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.

Settings for the InterfaceSalesOrderToCustomerEBFV2 service property:

Property Name	Value/Default Value	Property
Routing.SalesOrderOrchestrationEBSV2	http://<SOA_HOST>:<SOA_PO	CAVS Endpoint URI, when CAVS is

Property Name	Value/Default Value	Property
.InterfaceCustomerToFulfillment.CAVS.EndpointURI	RT>/AIAValidationSystemServlet/asyncresponsesimulator	enabled.
Routing.SalesOrderOrchestrationResponseEBSV2.InterfaceSalesOrderToCustomerResponse.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SalesOrderOrchestrationResponseEBSV2.InterfaceSalesOrderToCustomerResponse.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/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 aborts. Duration to wait for the asynchronous InterfaceCustomerToFulfillmentResponse callback from the SalesOrderOrchestrationResponseEBS service. If response is not received within the wait duration, a fault will occur.

Settings for the ProcessSalesOrderSiebelReqABCSImplV2 service property:

Property Name	Value/Default Value	Property
Default.SystemID	SEBL_01	Siebel system code (like SEBL_01, defined in BSR) 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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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&amp;SWEExtCmd=Execute&amp;UserName=<USER_NAME>&amp;Password	Siebel SWIOrderUpsert service endpoint location. This is a SOAP endpoint URL. If the request message contains the target URL, then that takes precedence.

Property Name	Value/Default Value	Property
	=<PASSWORD>	
Routing.SWIOrderUpsert.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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.
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) will be 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) will be invoked.
RMAOrderTypes	RMA	This property defines the value for RMA Order Types. List of code values that represent RMA order type. Affects the attribute mappings. This is a comma separated (no spaces) list of string values.

Settings for the ProcessQuoteSiebelReqABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	SEBL_01	Siebel system code (like SEBL_01, defined in BSR) 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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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.

Property Name	Value/Default Value	Property
Routing.SWIQuoteUpsert.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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&amp;SWEExtCmd=Execute&amp;UserName=<USER_NAME>&amp;Password=<PASSWORD>	Siebel SWIQuoteUpsert service endpoint location. This is a SOAP endpoint URL. If the request message contains the target URL, then that takes precedence.
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) will be 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) will be invoked

Settings for the UpdateSalesOrderEbizReqABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	EBIZ_01	Ebiz system code (like EBIZ_01, defined in BSR) 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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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.

Property Name	Value/Default Value	Property
Routing.GetSalesOrderEbizAdpater.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.GetSalesOrderEbizAdpater.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/esb/wsil/AIASystem/Ebiz/ABCS/GetSalesOrderEbizAdpater	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 or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.GetItemValidationOrgEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/GetItemValidationOrganizationEbizAdapter	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 or not during create order process
XRefCheck.TotalWaitTime	PT30S	Specifies the total wait time for which the service waits whether xref has been populated or not 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) will be 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) will be invoked.

Settings for the UpdateSalesOrderSiebelProvABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	SEBL_01	Siebel system code (like SEBL_01, defined in BSR) from which requests originate for this process.
Routing.SWIOrderUpsertService.SEBL_01.EndpointURI	http://<SIEBEL_HOST>/eai_enu/start.swe?SWEExtSource=SecureWebService&amp;SWEExtCmd=Execute&amp;UserName=<USER_NAME>&amp;Password=<PASSWORD>	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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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&amp;SWEExtCmd=Execute&amp;UserName=<USER_NAME>&amp;Password=<PASSWORD>	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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SalesOrderResponseEBSV2.U	http://<SOA_HOST>:<SOA_PO	CAVS Endpoint URI, when CAVS is

Property Name	Value/Default Value	Property
pdateSalesOrderResponse.CAVS.EndpointURI	RT>/AIAValidationSystemServlet/asyncresponserecipient	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) will be 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) will be 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) will be invoked
ABCSExtension.PostInvokeSBLOrderUpsertABM	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) will be invoked
ABCSExtension.PostInvokeSBLQuoteUpsertABM	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) will be 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) will be invoked

Settings for the CreateSalesOrderEbizProvABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	EBIZ_01	Ebiz system code (like EBIZ_01, defined in BSR) from which requests originate for this process
Routing.ProcessSalesOrderEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/ProcessSalesOrderEbizAdapter	Endpoint URI of the Ebiz adapter
Routing.ProcessSalesOrderEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS

Property Name	Value/Default Value	Property
		endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.ProcessSalesOrderEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.SalesOrderResponseEBSV2.CreateSalesOrderResponse.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SalesOrderResponseEBSV2.CreateSalesOrderResponse.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponserecipient	CAVS Endpoint URI, when CAVS is enabled.
RMAOrderTypes	RMA	This property defines the value for RMA Order Types. The list of code values that represent RMA order type. Affects the attribute mappings. This is a comma separated (no spaces) list of string values.
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) will be 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) will be 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) will be 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) will be invoked.
OrderSourceReference	COMMON/SourceSystemIdentifi	Property to decide OSR should hold the Common ID or Source System

Property Name	Value/Default Value	Property
	er Default=COMMON	Identifier.
DefaultOrderSourceId	28	Property to default the ORDER_SOURCE_ID if dvm lookup failure for quotes.
DefaultQuoteSourceId	29	Property to default the ORDER_SOURCE_ID if dvm lookup failure for quotes.

Settings for the CheckATPSalesOrderSiebelReqABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	SEBL_01	Siebel system code (such as SEBL_01, defined in BSR) 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) will be 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) will be invoked.
ABCSExtension.PostInvokeEBSCheckATPEBM	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) will be 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) will be invoked.
Routing.SalesOrderEBS.ProcessSalesOrderATPCheck.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the

Property Name	Value/Default Value	Property
		CAVS.EndpointURI property.
Routing.SalesOrderEBS.ProcessSalesOrderATPCheck.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.

Settings for the PaymentAuthorizationSalesOrderSiebelReqABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	SEBL_01	Siebel system code (like SEBL_01, defined in BSR) from which requests originate for this process
Routing.ReceivedPaymentEBS.ProcessCreditChargeAuthorization.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) will be invoked.
ABCSExtension.PreInvokeEBSPaymentAuthorizationEBM	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) will be invoked.
ABCSExtension.PostInvokeEBSPaymentAuthorizationEBM	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) will be 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) will be invoked.
Routing.ReceivedPaymentEBS.ProcessCreditChargeAuthorization.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.ReceivedPaymentEBS.ProcessCreditChargeAuthorization.CAVS.End	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServle	CAVS Endpoint URI, when CAVS is enabled.

Property Name	Value/Default Value	Property
pointURI	t/syncresponsesimulator	

Settings for the CreditCheckSalesOrderSiebelReqABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	SEBL_01	Siebel system code (like SEBL_01, defined in BSR) 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 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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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) will be invoked.
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) will be 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) will be 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) will be invoked.

Settings for the CalculateShippingChargeSalesOrderSiebelReqABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	SEBL_01	Siebel system code (such as SEBL_01, defined in BSR) 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 to the target service.
Routing.SalesOrderEBSV2.ProcessSalesOrderShippingChargeCalculation.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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) will be invoked.
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) will be 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) will be 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) will be invoked.

Settings for the ProcessSalesOrderATPCheckEbizProvABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	EBIZ_01	Ebiz system code (such as EBIZ_01, defined in BSR) from which requests

Property Name	Value/Default Value	Property
		originate for this process.
Routing.ProcessSalesOrderATPCheckEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/ProcessSalesOrderATPCheckEbizAdapter	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 to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.FetchATPScheduleSequenceEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/FetchATPScheduleSequenceEbizAdapter	Endpoint URI of the Ebiz adapter
Routing.FetchATPScheduleSequenceEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled
Routing.FetchATPScheduleSequenceEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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) will be 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) will be 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) will be invoked.
ABCSExtension.PostInvokeCheckATP	true/false Default = false	Governs whether ABCS Extension is

Property Name	Value/Default Value	Property
EbizAdapterABM		enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) will be invoked.

Settings for the ProcessCreditChargeAuthorizationEbizProvABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	EBIZ_01	Ebiz system code (like EBIZ_01, defined in BSR) from which requests originate for this process
Routing.ProcessCreditChargeAuthorizationEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/ProcessCreditChargeAuthorizationEbizAdapter	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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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) will be 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) will be 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) will be 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) will be invoked.

Settings for the ProcessCreditEligibilityEbizProvABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	EBIZ_01	Ebiz system code (like EBIZ_01, defined in BSR) from which requests originate for this process.
Routing.ProcessCreditEligibilityEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/ProcessCreditEligibilityEbizAdapter	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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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) will be invoked.
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) will be 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) will be 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) will be invoked.

Settings for the ProcessSalesOrderShippingChargeLogisticsProvABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	LOGIS_01	Logistics system code (like LOGIS_01, defined in BSR) from which requests originate for this

Property Name	Value/Default Value	Property
		process.
Routing.GLogService.LOGIS_01.EndpointURI	http://otm-dobson-55-oas.us.oracle.com/GC3/glog.webserver.servlet.umt.Login	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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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) will be 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) will be 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) will be 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) will be invoked.
Logistics.CalculateShippingCharge.TransmissionType	QUERY	Specifying Logistics web service. It is a query type. (This is a constant for rate enquiry)
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

Settings for the InterfaceSyncProductStructureEBF service property:

Property Name	Value/Default Value	Property
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 to the target service.
Routing.ItemCompositionEBS.SyncItemCompositionList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 to the target service.
Routing.ItemCompositionEBS.QueryItemCompositionList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 need to be synced. Default value is O2C that synchronizes the product structure.

Settings for the RequestProductStructureSiebelReqABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	SEBL_01	Siebel system code (like SEBL_01, defined in BSR) from which requests originate for this process
ABCSExtension.PreXformABMToEBM	true/false Default = false	Governs whether ABCS Extension is

Property Name	Value/Default Value	Property
ABM		enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) will be 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) will be invoked
Routing.ItemCompositionOrchestrationEBS.ProcessItemComposition.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 to the target service.
Routing.ItemCompositionOrchestrationEBS.ProcessItemComposition.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponserecipient	CAVS Endpoint URI, when CAVS is enabled.

Settings for the CreateProductEbizReqABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	EBIZ_01	Ebiz system code (like EBIZ_01, defined in BSR) to which data is synced into. 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>/event/AIASystem/Ebiz/ABCS/QuerySimpleItemAdapter	Endpoint URI of Ebiz Adapter
Routing.QuerySimpleItemAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.ItemEBS.SyncItemList.RouteT	true/false Default = false	Governs whether the service should route the message to the CAVS

Property Name	Value/Default Value	Property
oCAVS		endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 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 Bypassing the validation of Organization associated with item against OE:Item Validation Org. By Default 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) will be 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) will be invoked.
PUBLISH_ALL_PRODUCT	Y/N Default = N	Governs whether to sync both non-orderable and orderable products (when value is 'Y') or sync only non-orderable products while performing Item sync (when value is 'N')

Settings for the UpdateProductEbizReqABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	EBIZ_01	Ebiz system code (like EBIZ_01, defined in BSR) to which data is synced into. The system code value present in the EBM header of the incoming message takes

Property Name	Value/Default Value	Property
		precedence over this property value.
Routing.QuerySimpleItemUpdateAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/QuerySimpleItemUpdateAdapter	Endpoint URI of Ebiz Adapter
Routing.QuerySimpleItemUpdateAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
BYPASS_ITEMVALIDATIONORG_FLAG	Y/N Default =N	This property controls Bypassing the validation of Organization associated with item against OE:Item Validation Org. By Default 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) will be 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) will be invoked
Routing.ItemEBS.SyncItemList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 to the target service.
Routing.ItemEBS.SyncItemList.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponserecipient	CAVS Endpoint URI, when CAVS is enabled.
Routing.QuerySimpleItemUpdateAdapt	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/a	CAVS Endpoint URI, when CAVS is

Property Name	Value/Default Value	Property
er.CAVS.EndpointURI	syncresponserecipient	enabled.

Settings for the BulkLoadProductEbizReqABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	EBIZ_01	Ebiz system code (like EBIZ_01, defined in BSR) to which data is synced into. The System code value present in the EBM Header of the incoming message takes precedence over this property value.
Routing.QuerySimpleItemBulkLoadAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/QuerySimpleItemBulkLoadAdapter	Endpoint URI of Ebiz Adapter
Routing.QuerySimpleItemBulkLoadAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
BYPASS_ITEMVALIDATIONORG_FLAG	Y/N Default =N	This property controls Bypassing the validation of Organization associated with item against OE:Item Validation Org. By Default 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) will be 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) will be invoked
Routing.ItemEBS.SyncItemList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the

Property Name	Value/Default Value	Property
		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 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.

Settings for the SyncProductSiebelProvABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	SEBL_01	Siebel system code (like SEBL_01, defined in BSR) from which requests originate for this process
SIMPLE_PRODUCT_RELEASE_FLAG	Y/N Default =Y	This property controls the Releasing of Simple Product in Siebel. By Default Simple Product (That is Bundle product) in Siebel will be released.
COMPLEX_PRODUCT_RELEASE_FLAG	Y/N Default =N	This property controls the Releasing of Complex Product in Siebel. By Default Complex Product (That is Customizable product) in Siebel will not be released.
Routing.SWIProductIntegrationIORes.SEBL_01.EndpointURI	http://<SIEBEL_HOST>/eai_enu/start.swe?SWEExtSource=SecureWebService&amp;SWEExtCmd=Execute&amp;UserName=<USER_NAME>&amp;Password=<PASSWORD>	Endpoint URI of Siebel Service
Routing.SWIProductIntegrationIORes.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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.

Property Name	Value/Default Value	Property
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 to the target service.
Routing.ItemResponseEBSV2.SyncItemListResponse.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
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) will be 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) will be 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) will be 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) will be invoked.
Product.Source.PIP	MDM/O2C Default = O2C	Governs whether O2C or MDM is the product master

Settings for the SyncItemCompositionListSiebelProvABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	SEBL_01	Siebel system code (like SEBL_01, defined in BSR) from which requests originate for this process
Routing.SWIProductIntegration.SEBL_01.EndpointURI	http://<SIEBEL_HOST>/eai_enu/start.swe?SWEExtSource=SecureWebService&amp;SWEExtCmd=Execute&amp;UserName=<USER_NAME>&amp;Password=<PASSWORD>	Endpoint URI of Siebel Service
Routing.SWIProductIntegration.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 to the target service.
Routing.ItemCompositionResponseEBS.SyncItemCompositionListResponse.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 the Releasing of Simple Product in Siebel. By Default Simple Product (That is Bundle product) in Siebel will be released.
COMPLEX_PRODUCT_RELEASE_FLAG	Y/N Default =N	This property controls the Releasing of Complex Product in Siebel. Default Complex Product (That is Customizable product) in Siebel will

Property Name	Value/Default Value	Property
		not be 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) will be 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) will be 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) will be 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) will be invoked

Settings for the QueryItemCompositionListEbizProvABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	EBIZ_01	Ebiz system code (like EBIZ_01, defined in BSR) to which data is synced into.
PUBLISH_ALL_PRODUCT	Y/N Default =N	Governs whether the ABCS process should publish all child components. By default, the process Publishes only non-orderable 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) will be 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

Property Name	Value/Default Value	Property
		Extension process (defined along AIA ABCS Extension guidelines) will be 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) will be 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) will be 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) will be 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) will be 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) will be 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) will be invoked
Routing.ExplodeItemCompositionEbizAdapter.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/syncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.ExplodeItemCompositionEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/ExplodeItemCompositionEbizAdapter	Endpoint URI of Ebiz Adapter

Property Name	Value/Default Value	Property
Routing.ExplodeItemCompositionEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.QueryItemCompositionEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/QueryItemCompositionEbizAdapter	Endpoint URI of Ebiz Adapter
Routing.QueryItemCompositionEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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>/event/AIASystem/Ebiz/ABCS/QueryStructureItemCompositionEbizAdapter	Endpoint URI of Ebiz Adapter
Routing.QueryStructureItemCompositionEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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

Settings for the CreateAssetSiebelProvABCSImpl service property:

Property Name	Value/Default Value	Property
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 (like SEBL_01, defined in BSR) from which requests originate for this process

Property Name	Value/Default Value	Property
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) will be 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) will be invoked
Routing.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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&amp;SWEExtCmd=Execute&amp;UserName=<SIEBEL_UN>&amp;Password=<SIEBEL_UN>	Endpoint URI of Siebel Service

Settings for the UpdateAssetSiebelProvABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	SEBL_01	Siebel system code (like SEBL_01, defined in BSR) 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) will be invoked
ABCSExtension.PreProcessEBM	true/false Default = false	Governs whether ABCS Extension is

Property Name	Value/Default Value	Property
		enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) will be invoked
Routing.SWIAssetManagementIO.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.
Routing.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 or not. Default value is false, which does not route to CAVS. If set to true, it routes 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&amp;SWEExtCmd=Execute&amp;UserName=<SIEBEL_UN>&amp;Password=<SIEBEL_UN>	Endpoint URI of Siebel Service

Settings for the UpdateItemInstanceEbizReqABCSImpl service property:

Property Name	Value/Default Value	Property
Routing.QueryItemInstanceEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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>/event/AIASystem/Ebiz/ABCS/QueryItemInstanceEbizAdapter EBiz version 12.1.1 EndpointURI	Endpoint URI of Ebiz Adapter

Property Name	Value/Default Value	Property
	should be http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/QueryItemInstanceEbizR12VersionAdapter	
Routing.QueryItemInstanceOrderEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.QueryItemInstanceOrderEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/QueryItemInstanceOrderEbizAdapter	Endpoint URI of Ebiz Adapter
Default.SystemID	EBIZ_01	Ebiz system code (like EBIZ_01, defined in BSR) to which data is synced into.
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) will be 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) will be 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 or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.InstalledProductEBSV2.Updat	Default value = PRODUCTION	Governs whether the message is

Property Name	Value/Default Value	Property
eInstalledProductList.MessageProcessingInstruction.EnvironmentCode		routed to CAVS or to the specified target service. Default value is PRODUCTION, which routes to the target service.
Routing.InstalledProductEBSV2.UpdateInstalledProductList.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled
Routing.QueryItemInstanceEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.QueryItemInstanceEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/QueryItemInstanceEbizAdapter	Endpoint URI of Ebiz Adapter
Routing.QueryItemInstanceOrderEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.

Settings for the CreateItemInstanceEbizReqABCSImpl service property:

Property Name	Value/Default Value	Property
Routing.QueryItemInstanceEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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>/event/AIASystem/Ebiz/ABCS/QueryItemInstanceEbizAdapter EBiz version 12.1.1 EndpointURI should be http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/QueryItemInstanceEbizR12Versio	Endpoint URI of Ebiz Adapter

Property Name	Value/Default Value	Property
	nAdapter	
Routing.QueryItemInstanceOrderEbizAdapter.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.QueryItemInstanceOrderEbizAdapter.EBIZ_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/event/AIASystem/Ebiz/ABCS/QueryItemInstanceOrderEbizAdapter	Endpoint URI of Ebiz Adapter
Default.SystemID	EBIZ_01	Ebiz system code (like EBIZ_01, defined in BSR) to which data is synced into.
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) will be 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) will be 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.CreateInstalledProductList.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes 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 to the target service.

Property Name	Value/Default Value	Property
Routing.InstalledProductEBSV2.CreateInstalledProductList.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	CAVS Endpoint URI, when CAVS is enabled.

Settings for the SyncShipmentAdviceLsitSiebelProvABCSImpl service property:

Property Name	Value/Default Value	Property
Default.SystemID	Default value = SEBL_01	Siebel system code (such as SEBL_01, defined in BSR) from which requests originate for this process.
Routing.SWIOrderUpsertService.SEBL_01.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	Web service endpoint for the particular system instance.
Routing.SWIOrderUpsertService.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.SWIOrderUpsertService.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	CAVS endpoint URL with optional simulator ID included in the query string.
Routing.ShipmentAdviceResponseEBSV1.SyncShipmentAdviceListResponse.RouteToCAVS	true/false Default = false	Governs whether the service should route the message to the CAVS endpoint or not. Default value is false, which does not route to CAVS. If set to true, it routes to CAVS using the endpoint specified in the CAVS.EndpointURI property.
Routing.ShipmentAdviceResponseEBSV1.SyncShipmentAdviceListResponse.CAVS.EndpointURI	http://<SOA_HOST>:<SOA_PORT>/AIAValidationSystemServlet/asyncresponsesimulator	CAVS endpoint URL with optional simulator ID included in the query string.
ABCSExtension.PreXformEBMtoABMEBM	true/false Default = false	Enable/disable ABCS extension callout.
ABCSExtension.PreInvokeSBLOrderUpsertABM	true/false Default = false	Enable/disable ABCS extension callout.
ABCSExtension.PostInvokeSBLOrderUpsertABM	true/false Default = false	Enable/disable ABCS extension callout.
ABCSExtension.PostXformABMtoEBMEBM	true/false Default = false	Enable/disable ABCS extension callout.
ABCSExtension.PreInvokeShipmentAdviceEBS	true/false Default = false	Enable/disable ABCS extension callout.

Settings for the TransformAppContextEbizService service property:

Property Name	Value/Default Value	Property
TransformAppContextEbizService.Soap.EndpointURL	N/A	Endpoint URL for the service
TransformAppContextEbizService.EBIZ_01.ResponsibilityDVM	ORACLE_RESPONSIBILITY	DVM used to lookup 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	It is the responsibility of the application to send the system ID from which the request is being sent. If any requestor application fails to send this, AIA will pick 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

Settings for the TransformAppContextSiebelService service property:

Property Name	Value/Default Value	Property
TransformAppContextSiebelService.Soap.EndpointURL	N/A	Endpoint URL for the service
TransformAppContextSiebelService.DefaultUser	User1	Default Siebel user
TransformAppContextSiebelService.DefaultBusinessUnit	SiebelAdmin	Default Siebel Business Unit.

Performing Post-Installation Configurations

This section discusses the following post-installation configurations for the Oracle Product Hub PIP:

- Configuration required for Oracle Product Hub PIP and Order to Cash: Siebel CRM – EBS PIP Interoperability.
- Deploying services and creating grants to methods.
- Subscribe to Business Events.

Configuration for Oracle Product Hub PIP and Order to Cash: Siebel CRM – EBS PIP Interoperability

For more information, see AIA 2.5 PIP Functional Interoperability Configuration Guide.

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 will be 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.

Subscribing to the Business Events

To subscribe to Business Events:

1. Log in to as SYSADMIN/SYSADMIN.

   a. Navigate to Workflow Administrator, Web Applications Responsibility, Business Events.

   b. Query for business event oracle.apps.ego.item.publishItem, or oracle.apps.bom.structure.publishStructure.

   c. Click Subscription button.

   d. Enter the following details in the Subscription page:

```
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
```

   e. 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, refer to the *Oracle Application Integration Architecture: Installation and Upgrade Guide*.

Appendix A

These instructions are provided as an example of how to set up a data server for a non-Oracle database.

Configuring ODI-Based Initial Loads Against a Non-Oracle Target Database

Oracle Data Integrator (ODI) is a bulk-loading tool, used to load data between different systems. ODI is used in the Order to Cash: Siebel CRM – EBS PIP for the initial bulk load of Assets, Customers, Products, and Price Lists, using Oracle EBS as the source system and Siebel CRM as the target system.

ODI provides the flexibility to use source and target systems from a variety of database technologies like Oracle, MS-SQL, Sybase, DB2 and the like. 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, we have used the Customer bulk load as an example. Similar instructions can be followed for all of the other initial loads.

Frequently Used Terms

Here are the definitions of terms used in this section:

1. **Topology**: This is a GUI-based interface of ODI that makes physical connections to the data objects, such as Database, XML files and so forth

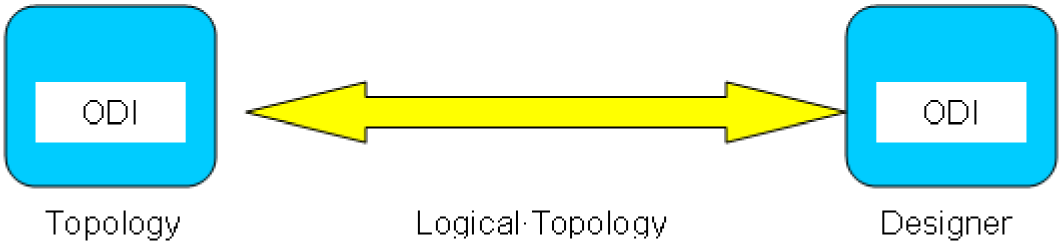
   a Physical Topology: This defines the connections to the database and file locations for XML files.

   This includes DB user, DB password, SID, Host, Port and so forth. The physical topology must be modified to point to the new Source and Target locations, if they are different from what is specified in deploy.properties.

Database
Physical·Topology
ODI

Physical topology

1. Logical Topology: This is the logical binding from ODI to the Data Models of the Designer.



Logical topology

2. Designer: This ODI interface defines the data models and the mappings between those data models.

   Data models are a logical representation of ODI objects, such as tables, files, views, and so forth

3. Operator: This interface displays the run time status of the bulk loads while they are being carried out.

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, 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 already 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.

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.
3. For this example, Siebel is using a database of technology ‘X’.
4. 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.

5. Now, create a new data server in the technology “X” tab.

   You can create a new data server by right clicking the tab and choosing the appropriate option from the menu.

6. Enter the appropriate database details (for example, DB username, password, URL, Port, and so forth).
7. Click the Test button to verify your database connection.

   After having successfully connected to the respective database, click OK or Apply, which will take you to a new window asking for the schema name 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.

8. 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. Make sure you give the same logical schema, or else it will not be able to connect to Siebel tables and will error out while executing bulk loads. Regarding O2C2, select the context as My_context from the menu and enter Logical schema name SEBL_SRVR_log_sch.

9. This completes the database setup.

Using the Designer

Follow these steps to use the Designer:

1. Log into 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 already selected.

3. Check for the context from the Reverse tab, if it matches with the context name of project in consideration.

   Click OK to make your changes.

4. Edit the appropriate interfaces using Siebel tables both as Sources and as Targets.
5. Import the “LKM SQL to SQL” and “IKM SQL to SQL Append” knowledge modules into the project.
6. In the interface screen, click 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 Siebel table is used as the source table and “IKM SQL to SQL Append” when the Siebel table is on the target side.

7. Save and exit the interface.

   Re-generate the scenario for command line execution.

Running the Customer Load—Example

This example is for the Customer initial load. Based on the previous steps, you can run the steps as follows:

1. Log into 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:

   a AIA Configuration Model

   b AIA Country Model

   c AIA Gender Model

   d AIA Initials Model

   e AIA State Model

   f OracleEbiz Model

   g Siebel Model

   h 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. Make sure 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:

   a 14_Ebiz_Cust_Eim_Account_Interface

   b 15_Ebiz_Cust_Eim_Contact_Interface

c 16_Ebiz_Cust_Eim_Addr_Per_Interface

d 17_Ebiz_Cust_Eim_Fn_Accnt1_Interfac

e 18_Siebel_CUSTOMERPARTY_PARTYID_XRF

f 19_Siebel_CUSTOMERPARTY_ACCOUNTID_X

g 20_Siebel_CUSTOMERPARTY_ADDRESSID_X

h 21_Sbl_CUSTMERPARTY_PARTYLOCATIONID

i 22_Sbl_CUSTOMERPARTY_LOCATIONREFID

j 23_Sbl_CUSTOMERPARTY_PARTYCONTACTID

k 24_Sbl_CUSTOMERPARYTY_CONTACTID

l 25_Sbl_CUSTOMERPARTY_CONTACT_COMMID

m 26_Sbl_CUSTOMERPARTY_ACCOUNT_COMMID

6. Expand the Knowledge Module under Customer_Bulk_Load and right click the Loading (LKM) option.

   Select the Import Knowledge Modules option. This opens a window that displays the current directory and knowledge modules present in that directory. Change the directory accordingly, 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 will 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:

   a OracleEbiz_to_XREF_to_EIM

   b 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. Now, you can run the packages OracleEbiz_to_XREF_to_EIM and Siebel_to_XREF.
12. You can also run the packages from the command line.

Index

Asset Management

- assumptions and constraints, 181
- core components, 191
- create assets flow, 183
- data requirements, 190
- integration services, 192
- Oracle EBS interfaces, 191
- overview, 180
- prerequisites, 180
- Siebel CRM interfaces, 190
- update assets flow, 187

Assumptions and Constraints

- Asset Management, 181
- ATP Check, 107
- Credit Check, 127
- Customer Management, 38
- for the PIP, 13
- initial data loads, 16
- Order Management, 143
- Payment Authorization, 135
- Price Lists, 92
- Product Management, 70
- Quotes integration, 99
- Shipping Charges, 117

ATP Check

- assumptions and constraints, 107
- core components, 111
- data requirements, 110
- integration services, 112
- Oracle EBS interfaces, 111
- overview, 105
- prerequisites, 107
- Siebel CRM interfaces, 111

BulkloadItemEbizEventConsumer, 90

BulkLoadProductEbizReqABCSImpl, 86

CalculateShippingChargeSalesOrderSiebelReqABCSImpl, 122

CheckATPSalesOrderSiebelReqABCSImpl, 113

Concurrent Processes, 221

Configuration Properties, 224

Configuring ODI-Based Initial Loads Against a Non-Oracle Target Database, 277

CreateAssetSiebelProvABCSImpl, 193

CreateItemEbizEventConsumer, 90

CreateItemInstanceEbizEventConsumer, 193

CreateItemInstanceEbizReqABCSImpl, 193

CreateProductEbizReqABCSImpl, 86

CreateSalesOrderEbizProvABCSImpl, 174

Creating Sales Orders, 146

Credit Check

- assumptions and constraints, 127
- core components, 130
- data requirements, 129
- integration services, 131
- Oracle EBS interfaces, 130
- overview, 125
- prerequisites, 127
- Siebel CRM interfaces, 130

CreditCheckSalesOrderSiebelReqABCSImpl, 132

Cross References for Siebel IDs and Oracle EBS Entities, 201

Cross-References, 213

Customer Management

- assumptions and constraints, 38
- core components, 58
- data requirements, 55
- integration services, 59
- merging accounts, 49
- merging parties, 52
- overview, 37
- prerequisites, 38
- Siebel CRM interfaces, 56
- synchronizing customer accounts, 46
- synchronizing new customer accounts, 40
- updating and synchronizing customer accounts, 43

CustomerPartyEBS, 131

CustomerPartyEBSV2, 60

CustomerPartyResponseEBSV2, 60

Domain Value Maps, 216

Error handling, 223

GetSalesOrderEbizAdapter, 177

Implementing the Order to Cash

- Siebel CRM – EBS Process Integration Pack
  - configuring the Payment Authorization flow, 222
  - creating Oracle EBS system profiles, 219
  - describing domain value maps, 216
  - enabling Oracle EBS events, 218
  - error handling, 223
  - identifying cross-references, 213
  - overview, 195
  - scheduling concurrent processes, 221
  - setting a property in OTM, 221
  - setting configuration properties, 224
  - setting up cross references for Siebel IDs and Oracle EBS entities, 201
  - setting up inventory locations, business units and organizations, 199

Initial Data Loads

- asset data, 33
- assumptions and constraints, 16
- customer data, 18
- deploying ODI repository components, 16
- non-Oracle databases, 277
- overview, 14
- prerequisites, 15
- price lists data, 27
- product data, 24

InstalledProductEBSV2, 192

InterfaceCustomerToFulfillmentEBF, 61

InterfaceSalesOrderToCustomerEBFV2, 171

InterfaceSalesOrderToFulfillmentEBF, 169

InterfaceSyncProductStructureEBF, 87

ItemCompositionEBS, 88

ItemCompositionOrchestrationEBS, 87

ItemCompositionResponseEBSV2, 85

ItemEBSV2, 85

ItemResponseEBSV2, 85

MergeAccountEbizEventConsumer, 68

MergeAccountEbizReqABCSImpl, 63

MergePartyEbizEventConsumer, 68

MergePartyEbizReqABCSImpl, 65

Oracle EBS events, 218

Order Management

  - sales order revision (Siebel CRM initiated), 157
  - sales order cancellation, 160

- assumptions and constraints, 143
- core components, 164
- creating sales orders, 146
- integration services, 165
- Oracle EBS interfaces, 163
- overview, 142
- prerequisites, 143
- Siebel CRM interfaces, 162
- updating sales orders (Oracle EBS initiated), 151

Order to Cash

- Siebel CRM – EBS Overview, 8
- Siebel CRM – EBS PIP
  - configuring participating applications for, 196
- Siebel CRM – EBS Solution Assumptions and Constraints, 13

Organizations, 199

Overview, 8

Payment Authorization

- assumptions and constraints, 135
- configuring, 222
- core components, 139
- data requirements, 138
- integration services, 140
- Oracle EBS interfaces, 139
- overview, 133
- prerequisites, 135
- Siebel CRM interfaces, 138

PaymentAuthorizationSalesOrderSiebelReqABCSImpl, 141

Price Lists

- assumptions and constraints, 92
- loading incremental, 93
- overview, 91
- prerequisites, 92

ProcessCreditChargeAuthorizationEbizProvABCSImpl, 141

ProcessCreditEligibilityEbizProvABCSImpl, 132

ProcessQuoteSiebelJMSConsumer, 103

ProcessQuoteSiebelJMSProducer, 103

ProcessQuoteSiebelReqABCSImpl, 104

ProcessSalesOrderATPCheckEbizProvABCSImpl, 113

ProcessSalesOrderEbizAdapter, 177

ProcessSalesOrderShippingChargeLogisticsProvABCSImpl, 122

ProcessSalesOrderSiebelJMSConsumerV2, 173

ProcessSalesOrderSiebelJMSProducerV2, 173

ProcessSalesOrderSiebelReqABCSImplV2, 174

Product Management

- assumptions and constraints, 70
- core components, 83
- create items flow, 74
- data requirements, 81
- integration services, 84
- Oracle EBS Interfaces, 57, 82
- overview, 69
- prerequisites, 70
- Siebel CRM interfaces, 81
- synchronizing BOMs, 77
- synchronizing items, 73
- update items flow, 76

QueryCustomerPartyListSiebelProvABCSImplV2, 61, 62

QueryItemCompositionListEbizProvABCSImpl, 88

QueryItemInstanceEbizAdapter, 192

Quotes

- assumptions and constraints, 99

- core components, 101
- data requirements, 100
- integration flow, 99
- integration services, 102
- Oracle EBS interfaces, 101
- overview, 98
- prerequisites, 98
- Siebel CRM interfaces, 100

ReceivedPaymentEBS, 140

RequestProductStructureSiebelJMSConsumer, 89

RequestProductStructureSiebelJMSProducer, 89

RequestProductStructureSiebelReqABCSImpl, 87

Running the Customer Load, 280

SalesOrderEBS, 112, 121

SalesOrderEBSV2, 166, 168

SalesOrderOrchestrationEBSV2, 167

Shipping Charges

- assumptions and constraints, 117
- core components, 120
- data requirements, 119
- integration services, 121
- OTM interfaces, 120
- overview, 115
- prerequisites, 117
- Siebel CRM interfaces, 120

Siebel CRM Integration Pack for Oracle Order Management: Order to Cash 2.5, 8

- introducing, 8

Solution Assumptions and Constraints, 13

SyncCustomerPartyListEbizAdapter, 68

SyncCustomerPartyListEbizEventConsumer, 67

SyncCustomerPartyListEbizProvABCSImpl, 63

SyncCustomerPartyListEbizReqABCSImpl, 62

SyncCustomerPartyListSiebelProvABCSImpl, 62

SyncItemCompositionListSiebelProvABCSImpl, 89

SyncProductSiebelProvABCSImpl, 86

System Profiles, 219

UpdateAssetSiebelProvABCSImpl, 194

UpdateCustomerPartySiebelReqABCSImplV2, 61

UpdateCustomerSiebelJMSConsumerV2, 66, 67

UpdateCustomerSiebelJMSProducerV2, 66

UpdateItemEbizEventConsumer, 90

UpdateItemInstanceEbizEventConsumer, 193

UpdateItemInstanceEbizReqABCSImpl, 193

UpdateProductEbizReqABCSImpl, 86

UpdateSalesOrderEbizEventConsumer, 175

UpdateSalesOrderEbizReqABCSImpl, 176

UpdateSalesOrderSiebelProvABCSImpl, 176

Using the Designer, 280

Using the Topology Manager, 279