Oracle® Application Integration Architecture

Oracle Financial Operations Control Integration Pack for Oracle Retail Merchandising Suite and E-Business Suite Financials Implementation Guide

Release 2.5

E16266-04

March 2013

The Oracle Retail Merchandise Integration Pack for E-Business Financials provides integration to a robust enterprise financial system to complement the Oracle Retail Merchandising system in a retail customer environment.

Oracle Application Integration Architecture Oracle Financial Operations Control Integration Pack for Oracle Retail Merchandising Suite and E-Business Suite Financials Implementation Guide, Release 2.5

E16266-04

Primary Author: Oracle Corporation

Contributing Author:

Contributor:

Contents

Preface

This preface discusses:

- Oracle Application Integration Architecture - Foundation Pack: Core Infrastructure Components Guide
- 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 Process Integration Packs
- Additional resources

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

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

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

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

The *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.
- 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 AIA services to work with the Composite Application Validation System (CAVS).
- Configure Oracle AIA processes to be eligible for error handling and logging.
- Extend enterprise business objects.

In addition, this book provides, Application Integration Architecture naming standards.

Oracle Application Integration Architecture Process Integration Packs

A process integration pack (PIP) is a prebuilt set of integrated orchestration flows, application integration logic, and extensible enterprise business objects and services required to manage the state and 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 to rapidly implement a discreet business process.

Additional Resources

The following resources are available:

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

Chapter 1: Understanding the Oracle Financial Operations Control Integration Pack for Oracle Retail Merchandise Operations Management and E-Business Suite Financials 2.5

This chapter provides an overview of the Oracle Financial Operations Control Integration Pack for Oracle Retail Merchandise Operations Management and E-Business Suite Financials and discusses:

- Key benefits
- Participating applications
- Retail Sales Financial business process flow
- Retail Inventory Financial business process flow
- Retail Procure to Pay business process flow
- Solution assumptions and constraints

Overview

The Oracle Retail Merchandise Integration Pack for E-Business Financials provides integration to a robust enterprise financial system to complement the Oracle Retail Merchandising system in a retail customer environment.

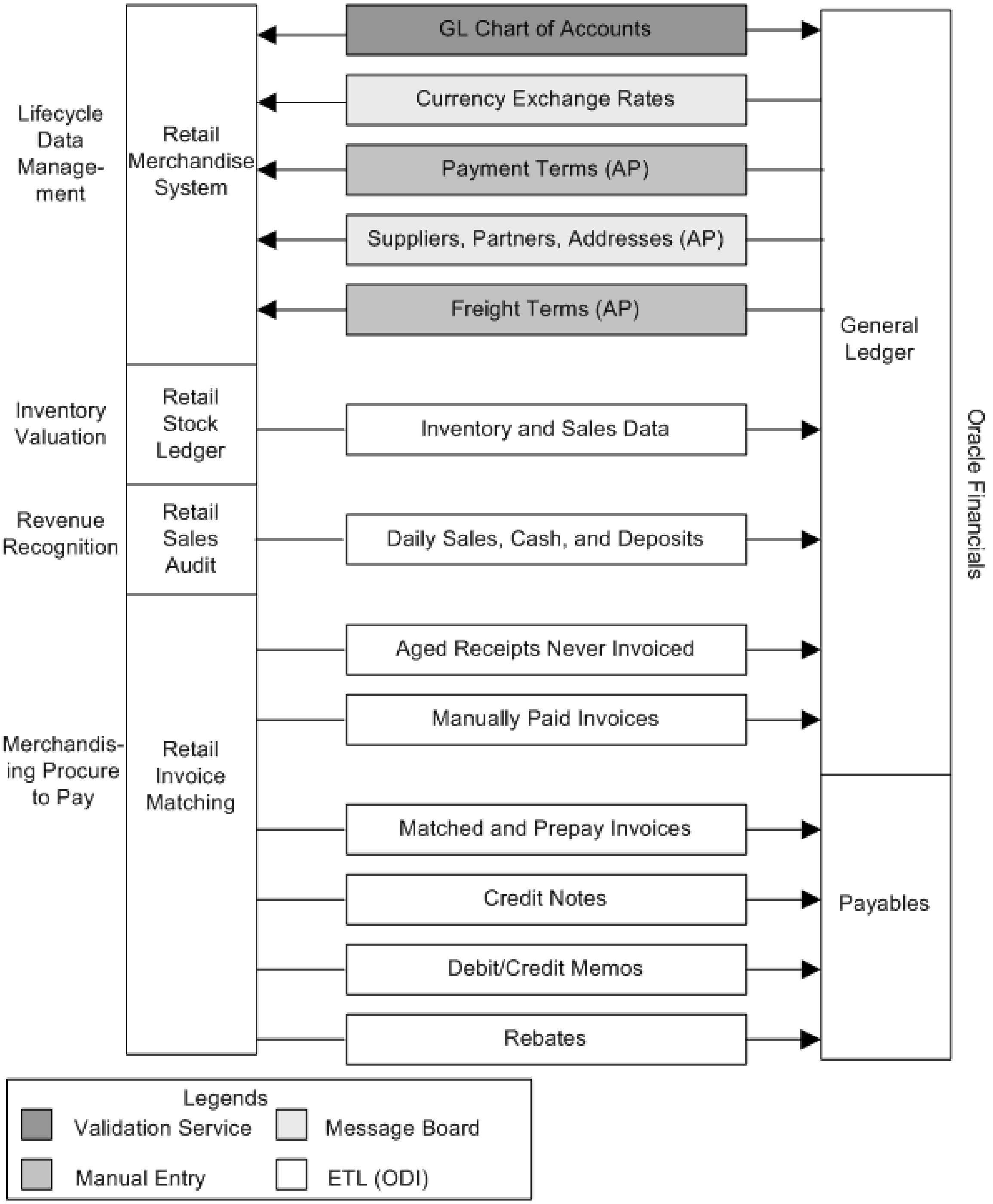
This Process Integration Pack (PIP) includes these four processes:

- Life Cycle Data Management: This process provides the data synchronization for the initial load prior to implementation and incremental data creation and maintenance after implementation. This process synchronizes suppliers and currency exchange rates from the E-Business applications to the Oracle Retail Merchandising System (RMS). Oracle Payables is the source of suppliers, payment terms, and freight terms. Because the freight terms and payment terms are static in nature and their volume is very low, they are synchronized between the two systems manually. This process enables users of the Oracle Retail Merchandise suite to carry out functions with data that is shared with the E-Business applications throughout the life cycle of creations and updates of the data.
- Inventory Valuation (Retail stock ledger): This process enables the posting of accounting entries generated from transactions that change the value of sellable products at a retailer’s inventory locations (stores and warehouses) to the appropriate ledgers from Oracle Retail Merchandising – stock ledger to Oracle General Ledger (GL). This process records the financial impact of changes in the sellable inventory in store and warehouse locations. Valuation of sellable inventory in the stores and warehouses is based on the processing of transactions for movement, pricing, costing, and sale of the inventory. This

valuation is captured and processed in Oracle Retail stock ledger. These transactions include sales, shipments from warehouse to store, store receipts, store transfers, returns to vendors, price changes, stock counts, and shrinkage due to theft or damage.

- Retail Revenue Recognition: This process enables the posting of accounting entries generated from sales and returns transactions from the retailer's stores for revenue and cash reconciliation to the appropriate ledgers. In this process, the data flows from Oracle Retail Sales Audit (ReSA) to Oracle GL. This process records the financial impact of sale/return, cash reconciliation, and void transactions from stores. The Revenue Recognition process begins when store transactions (sales and returns) are processed by ReSA. For each store transaction, ReSA generates the appropriate accounting entries to be posted to Oracle GL. Each accounting entry has a valid account code segment combination based on the transaction type, business unit, and location (store or warehouse).
- Retail Merchandising Procure to Pay: This process begins with the Oracle Retail Invoice Matching (ReIM) application. Invoices from suppliers for retail merchandise are matched to the original purchase order (PO) for the merchandise and the receipt of the merchandise by the retailer. A proper match of invoice, PO, and receipt trigger the payment authorization of the supplier's invoice. Invoices may be authorized for payment prior to receipt of goods for which prepayment is required. When the authorization for payment is generated, the appropriate accounting distribution is also generated to support the payment authorization. The Retail Merchandise Procure to Pay integration automates the processing of invoice payments, adjustments, and write-offs from ReIM to Oracle Payables and GL. Other accounting transactions are generated from ReIM to write off aged receipts that were never invoiced and to post accounting distribution for manually paid or prepaid invoices after receipt.

This diagram illustrates the Oracle Retail to Oracle Financials process flows:



Oracle Retail to E-Business Financials PIP process flow

This PIP does not synchronize chart of accounts from Oracle GL to Oracle Retail. Chart of accounts are combinations of account code segments. Because transaction types are defined and assigned combinations of code segments for proper handling of the financial impacts in Oracle Retail, the code combinations are validated by a service provided by the Oracle GL. This ensures that the accounting entries generated by the transactions are valid when they are posted to Oracle GL.

Key Benefits

Here are the key benefits of this PIP:

- This integration is not a point-to-point integration between the E-Business Suite and Oracle Retail applications. This PIP implementation is independent of version of integrated applications. An Application Integration Architecture (AIA) layer serves as an intermediate thin layer of application between E-Business and Oracle Retail. This integration remains synchronized with the new releases of the edge applications.
- Audited transaction data is exported to the E-Business Financial applications days before the traditional audit process permits. The Financials applications can use this timely data in a proactive manner, which results in increased productivity and operational efficiencies.
- Total cost of ownership for Oracle and its customers is reduced.

Participating Applications Overview

This section provides an overview of these applications participating in the process integration:

- Oracle Retail Merchandising System
- Oracle Retail Sales Audit
- Oracle Retail Invoice Matching
- Oracle Payables
- Oracle General Ledger

Oracle Retail Merchandising System

Oracle Retail Merchandising (RMS) is an integrated solution for global retailing. This solution enables retailers to better manage, control, and perform crucial day-to-day merchandising activities. From new product introduction to inventory management, RMS provides retailers with a complete end-to-end solution and is the most comprehensive and integrated solution for global retailing.

For more information, see the Oracle Retail Merchandising System User Guide.

Oracle Retail Sales Audit

Oracle Retail Sales Audit (ReSA) provides retailers with a flexible tool that evaluates and ensures accuracy and completeness of point of sale (POS) data. Realtime access to this audited sales data ensures integrity of information throughout the retail enterprise. With a highly configurable sales audit application, the retailer can maintain existing business practices while providing for future options as the operations grow and change.

ReSA enables retailers to receive POS transaction data, cleanse it, and export the data to the Oracle Merchandising system and the Oracle Retail Data Warehouse. By providing corporate control and visibility to sales audit information, ReSA enables retailers to make better decisions to improve merchandise operations and transform the economics of their business.

For more information, see the Oracle Retail Sales Audit User Guide.

Oracle Retail Invoice Matching

Oracle Retail Invoice Matching (ReIM) is a market-leading solution for retailers who need an automated application to better manage reconciliation and payment of purchase orders. This advanced solution enables account payables teams to resolve discrepancies on invoices quickly before payments are made. A highly automated, multidimensional matching engine minimizes time spent on manual reviews. Automated routing provides an effective method to ensure that accurate information is delivered to the right internal teams for resolution and compliance controls.

For more information, see the Oracle Retail Invoice Matching User Guide.

Oracle Payables

Oracle Payables provides automated invoice and payment processing to ensure timely and accurate payment for goods and services. Best-practice business processes match purchase orders, receipts, and invoices and provide online approvals to identify exceptions and increase control over disbursements.

Oracle Payables delivers built-in controls to help an enterprise meet regulatory requirements, enforce compliance, reduce risk, and implement due-diligence best practices reducing cycle times and errors. Other features include a flexible, user-defined system setup, extensive vendor maintenance, digital signatures, financials sanction validation, and powerful inquiry and analytical capabilities.

For more information, see the Oracle Payables User’s Guide.

Oracle General Ledger

Oracle General Ledger (GL) offers a fully automated close and consolidation solution for legal and management reporting, including support for Generally Accepted Accounting Principles (GAAP) and International Financial Reporting Standards (IFRS). Transactions are automatically processed and validated according to the best-practice business processes and control settings. In addition, an enterprise can proactively control expenditures by automatically checking spending requests against budget. With realtime reporting and information access, an enterprise can achieve complete visibility into financial results.

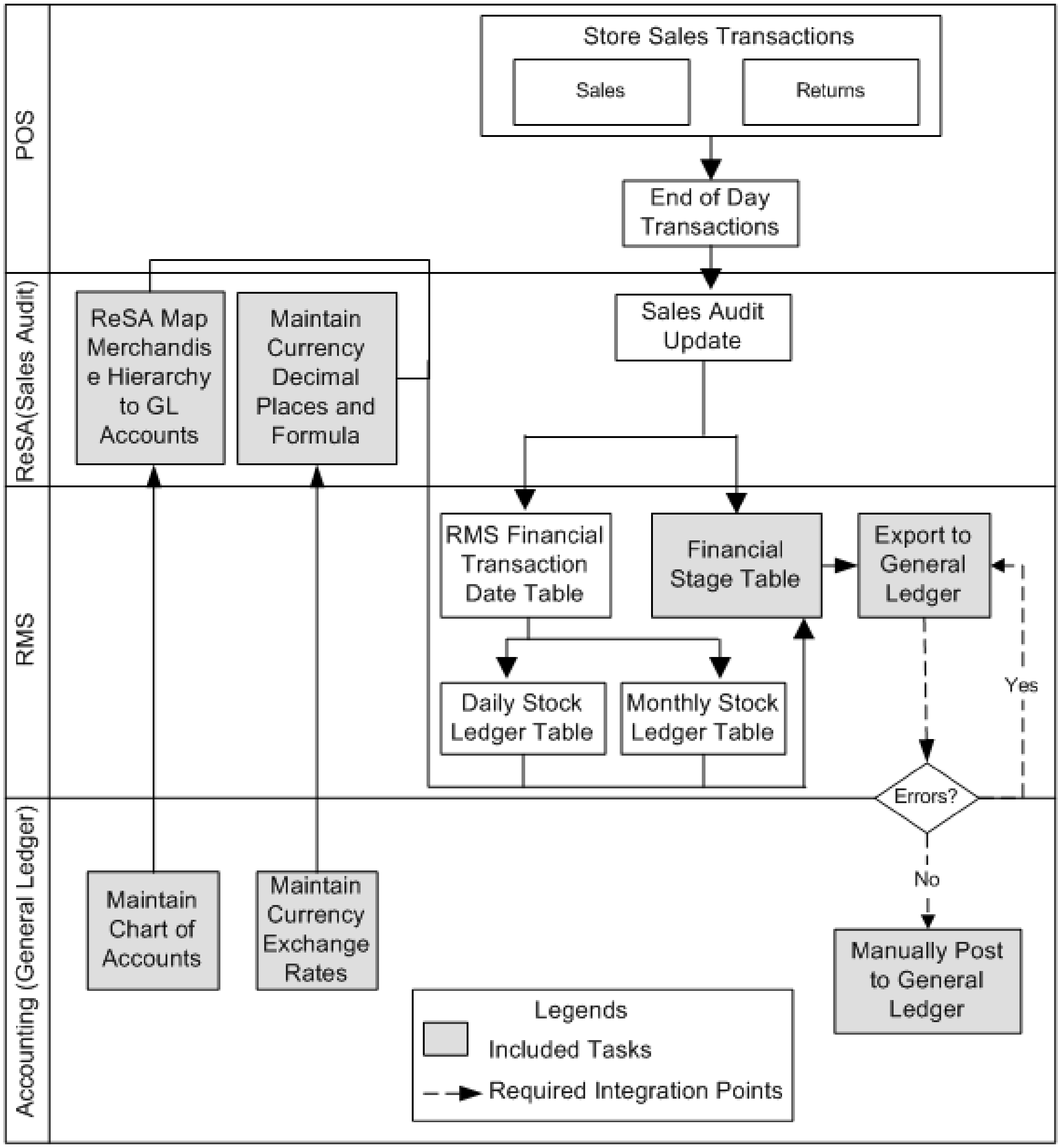
For more information, see the Oracle General Ledger User’s Guide

Retail Sales Financial Business Process Flow

The Retail Sales Financial business process consists of the Post Channel Sales, Cash, and Deposits from ReSA to Oracle GL integration flow.

This diagram illustrates the Retail Sales Financial business process flow:

Retail Sales Financial Management (Sales Audit)


POS
ReSA(Sales Audit)
RMS
Accounting (General Ledger)
Store Sales Transactions
Sales
Returns
End of Day Transactions
Sales Audit Update
ReSA Map Merchandise Hierarchy to GL Accounts
Maintain Currency Decimal Places and Formula
RMS Financial Transaction Date Table
Financial Stage Table
Export to General Ledger
Daily Stock Ledger Table
Monthly Stock Ledger Table
Errors?
Yes
No
Manually Post to General Ledger
Maintain Chart of Accounts
Maintain Currency Exchange Rates
Legends
Included Tasks
Required Integration Points

Retail Sales Financial business process flow

ReSA sends summarized sales audit information to Oracle GL for the Sales Journal. The sales audit information includes channel sales, cash, and deposits. The ReSA Export processes select and format corrected and preaudited data from the ReSA database so that it can be sent to E-Business Financials.

ReSA includes programs to automatically extract the required totals data and to format it to generic data files from a financial staging table for import into Oracle GL. Sales audit data from ReSA is also posted directly to the RMS stock ledger and can be integrated into Oracle GL through the stock ledger to the financial staging table and the accounting entry table. Before data is imported into Oracle GL, a batch process writes balanced records to the financial staging table using the appropriate General Ledger account combinations (maintained in Cross Reference tables in ReSA) and the Currency Exchange Rates.

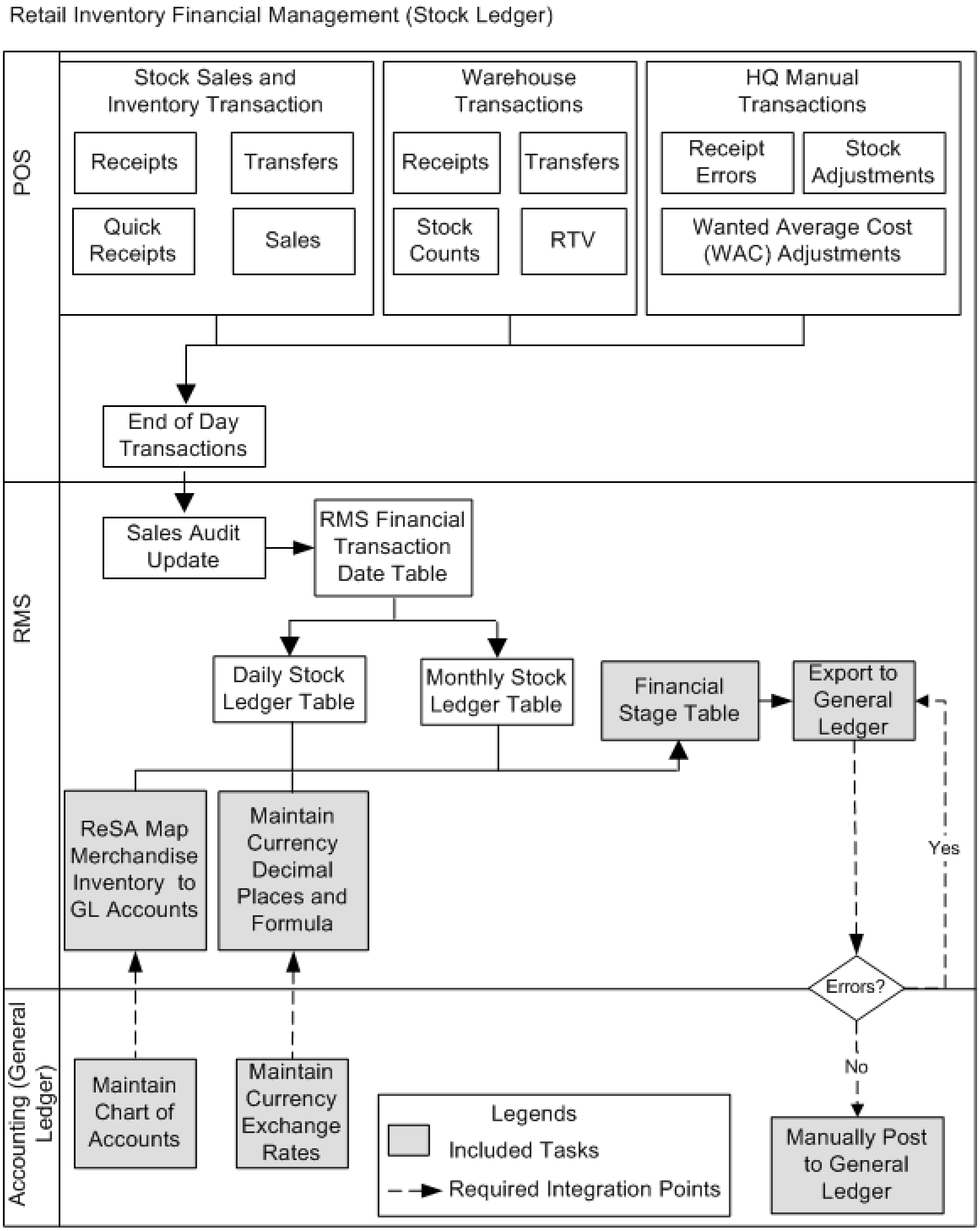
For journal entries, an ODI process is used to load the accounting entries into the GL_Interface table. Then a concurrent request, Journal Import, is run to move the journal entries into the GL transaction tables.

Retail Inventory Financial Business Process Flow

The Retail Inventory Financial business process consists of these integration flows:

- Post stock ledger from RMS to Oracle GL.
- Post write-offs (aged receipts, not invoiced to ledger) from RMS to Oracle GL.

This diagram illustrates the Retail Inventory Financial business process flow:


Retail Inventory Financial Management (Stock Ledger)
POS
Stock Sales and Inventory Transaction
Receipts
Transfers
Quick Receipts
Sales
Warehouse Transactions
Receipts
Transfers
Stock Counts
RTV
HQ Manual Transactions
Receipt Errors
Stock Adjustments
Wanted Average Cost (WAC) Adjustments
End of Day Transactions
RMS
Sales Audit Update
RMS Financial Transaction Date Table
Daily Stock Ledger Table
Monthly Stock Ledger Table
Financial Stage Table
Export to General Ledger
Yes
ReSA Map Merchandise Inventory to GL Accounts
Maintain Currency Decimal Places and Formula
Errors?
Accounting (General Ledger)
Maintain Chart of Accounts
Maintain Currency Exchange Rates
No
Legends
Included Tasks
Required Integration Points
Manually Post to General Ledger

Oracle Retail Inventory Financials business process flow

The stock ledger in RMS records financial results of the merchandising processes that occur in the Retail system, such as buying, selling, price changes, transfers, and so on. All of these transactions are recorded in the RMS stock ledger and rolled up to the subclass or location level for days, weeks, and months. Daily and period-based financial information is scheduled to be loaded into the E-Business Suite Financials. RMS sends three levels of stock ledger information to Oracle GL:

- Monthly – no access to detailed reference information
- Daily by subclass, class, or department – no access to detailed reference information.
- Daily by transaction

The stock ledger transactions to be loaded into E-Business Suite Financials are placed on the financial staging table through the use of table triggers or batch, by means of the appropriate General Ledger account combinations (maintained in the RMS cross-reference table in Oracle Retail) and the currency exchange rates.

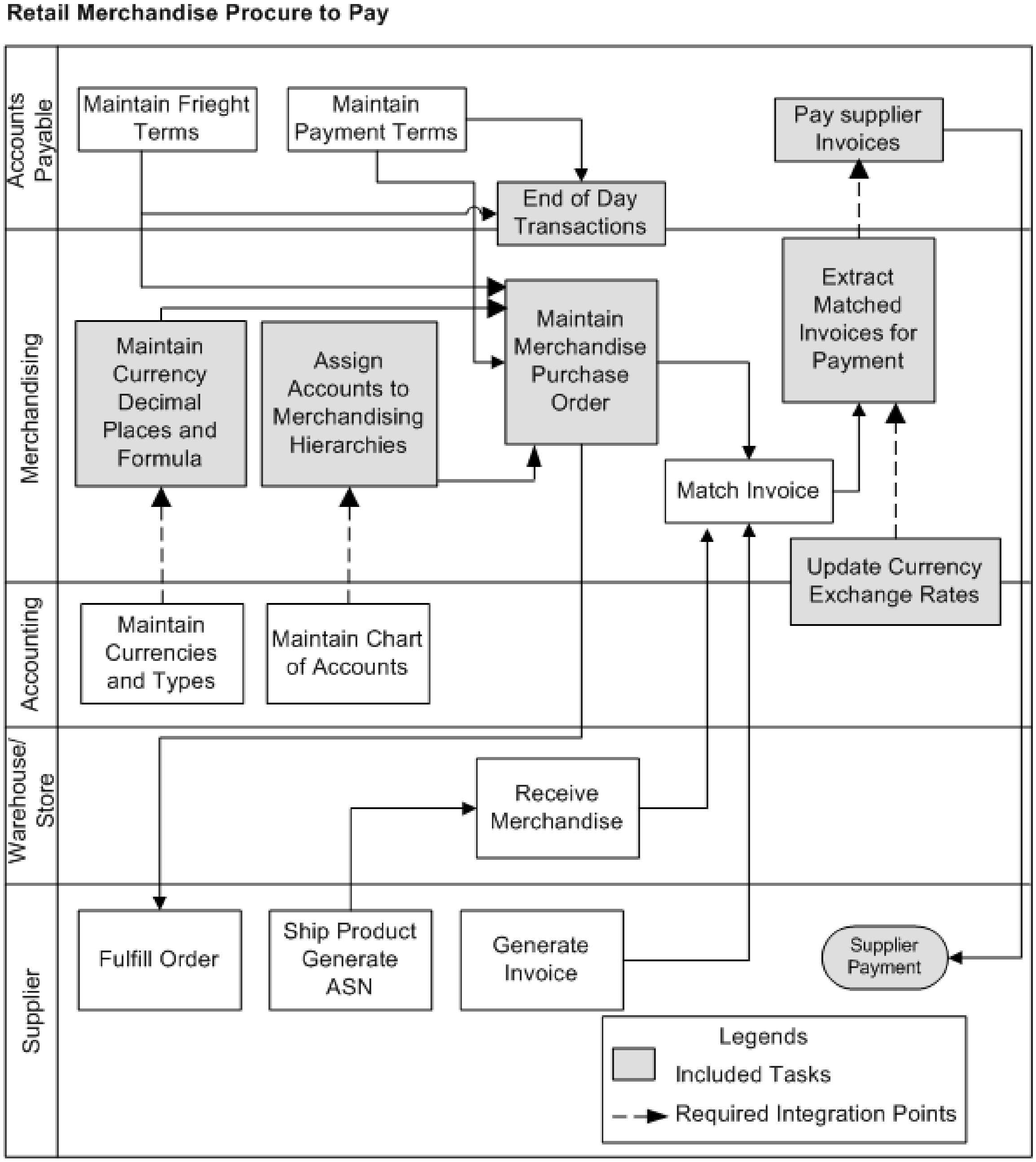
For journal entries, an ODI process is used to load the accounting entries into the GL_Interface table. Then a concurrent request, Journal Import, is run to move the journal entries into the GL transaction tables.

Retail Merchandise Procure to Pay Business Process Flow

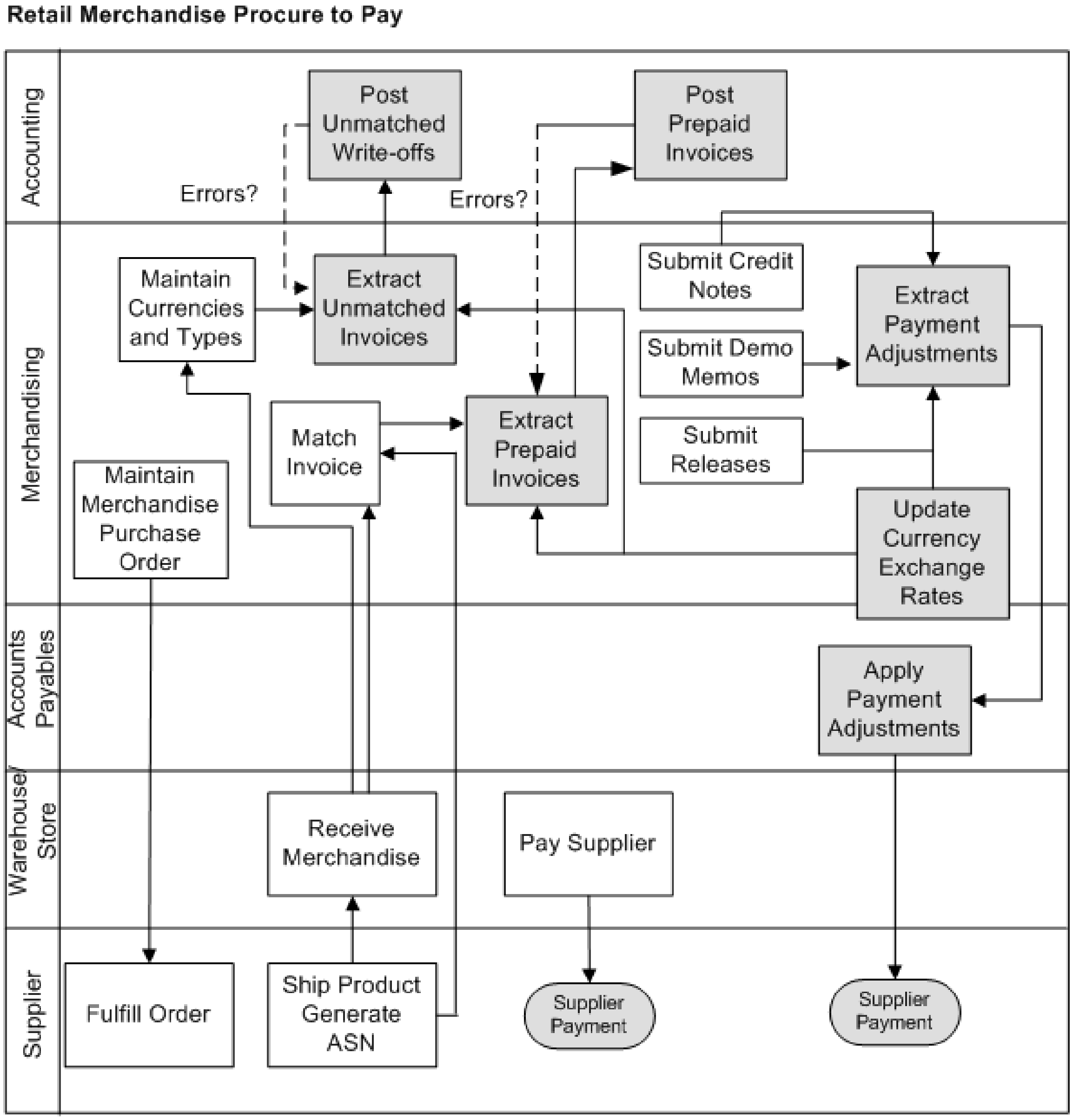
The Retail Merchandise Procure to Pay business process consists of these integration flows:

- Post matched prepaid invoices from ReIM to Oracle GL.
- Post manually matched paid Invoices from ReIM to Oracle GL.
- Post receipt write-offs from ReIM to Oracle GL.
- Post matched invoices for payment from ReIM to Oracle Payables.
- Post credit notes (matched or unmatched) for payment adjustment from ReIM to Oracle Payables.
- Post debit or credit memos for payment adjustment from ReIM to Oracle Payables.
- Post rebates for payment adjustment from ReIM to Oracle Payables.
- Post unmatched invoices for prepayment from ReIM to Oracle Payables.

This diagram illustrates the Retail Merchandise Procure to Pay business process flow:


Retail Merchandise Procure to Pay
Accounts Payable
Merchandising
Accounting
Warehouse/ Store
Supplier
Maintain Frieght Terms
Maintain Payment Terms
End of Day Transactions
Pay supplier Invoices
Extract Matched Invoices for Payment
Maintain Currency Decimal Places and Formula
Assign Accounts to Merchandising Hierarchies
Maintain Merchandise Purchase Order
Match Invoice
Update Currency Exchange Rates
Maintain Currencies and Types
Maintain Chart of Accounts
Receive Merchandise
Fulfill Order
Ship Product Generate ASN
Generate Invoice
Supplier Payment
Legends
Included Tasks
Required Integration Points

Retail Merchandise Procure to Pay business process flow (1 of 2)



Retail Merchandise Procure to Pay business process flow diagram (2 of 2)

The Retail Merchandise Procure to Pay business process flows enable posting of matched invoices, matched credit notes, debit and credit memos, rebates, and unmatched invoices for prepayment from ReIM to Oracle Payables.

For payables invoices are placed in the AP Interface tables. Then a concurrent request Payables Open Invoice Import is run to move the payables invoice into the payables transaction tables.

Solution Assumptions and Constraints

These are the assumptions and constraints for this PIP:

Assumptions:

1. The E-Business Suite applications are implemented prior to the implementation of this PIP.
2. Oracle Retail manually creates and stores the valid charts of accounts in the appropriate GL Cross Reference tables (ReSA, RMS, and ReIM).
3. The Retail stock ledger supports multiple currencies. All transaction-level information is stored in the local currency of the store or warehouse where the transaction occurred.
4. During the initial load for currency, historical and current rates are passed to Retail.
5. Oracle Retail sends the accounting date and the transaction date with its transactions. These dates should not be changed or manipulated in E-Business Suite Financials.
6. Accounting entry errors that are found from accounting entries are handled manually on both the Oracle Retail and E-Business side.
7. Use or sales tax accounting information is passed as part of the accounting entries between Oracle Retail and E-Business Financials.
8. Value-added tax (VAT) is calculated in Oracle Retail. VAT calculation is passed as a part of the accounting entry.
9. Oracle Retail stock ledger determines the valuation of inventory for merchandise being directly procured. This information is passed to E-Business Financials as the accounting entries.
10. RMS, through the Retail stock ledger, provides E-Business Financials with the value of ending inventory at cost using the method that the retailer indicates (cost method or retail method of accounting) by means of an adjusting entry.
11. Accounting entries need to be manually posted to general ledger.
12. Both E-Business Suite Financials and Oracle Retail support multiple organizations in one application instance.

Constraints

1. Customers switching from one financials application is not compatible with this PIP

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

Chapter 2: Life Cycle Management for Reference Data

This chapter discusses these process integrations:

- Currency exchange rate integration
- Suppliers information integration

Currency Exchange Rate Integration

This section provides an overview of the process integration for initial loading and incremental synchronization of currency exchange rates between Oracle General Ledger (GL) and the Oracle Retail Merchandising System (RMS) and discusses:

- Currency exchange rate integration details
- Oracle GL interfaces
- Oracle Retail Integration Bus (RIB) interfaces
- Data requirements
- Core Application Integration Architecture (AIA) components
- Integration services

Overview

Currency exchange rate is the reference information used in the translation of monetary values from one currency to another. The exchange rate expresses the value of one currency in terms of another. The process integration for currency exchange rates enables you to use Oracle Financials as an accounting engine and Oracle Retail for sales audit and stock ledger transactions.

The process integration for currency exchange rates supports these integration flows:

- Load initial currency exchange rate from Oracle GL to RMS: Enables the loading of all historical, current and future effective dated currency exchange rates from Oracle GL to RMS for a new instance (logical or physical) of RMS.
- Incremental creation and updates of currency exchange rates from Oracle GL to RMS: Enables the synchronization of incremental creation and updates of the historical, current and future effective dated currency exchange rates from Oracle GL to RMS.

This integration is not a point-to-point integration between Oracle GL and RMS. An AIA layer serves as an intermediate thin layer of application between Oracle GL and RMS. As a part of the currency exchange rates integration, Oracle GL sends the currency exchange rates to the AIA layer and the AIA layer delivers the information to RMS. The AIA layer performs message filtering, message transformation, and message routing. E-Business Suite triggers the integration by invoking SyncCurrencyExchangeListEbizReqABCSImpl with corresponding ABM's. If the program fails, then an error displays.

Prerequisites

These are the prerequisites for the process integration for currency exchange rate:

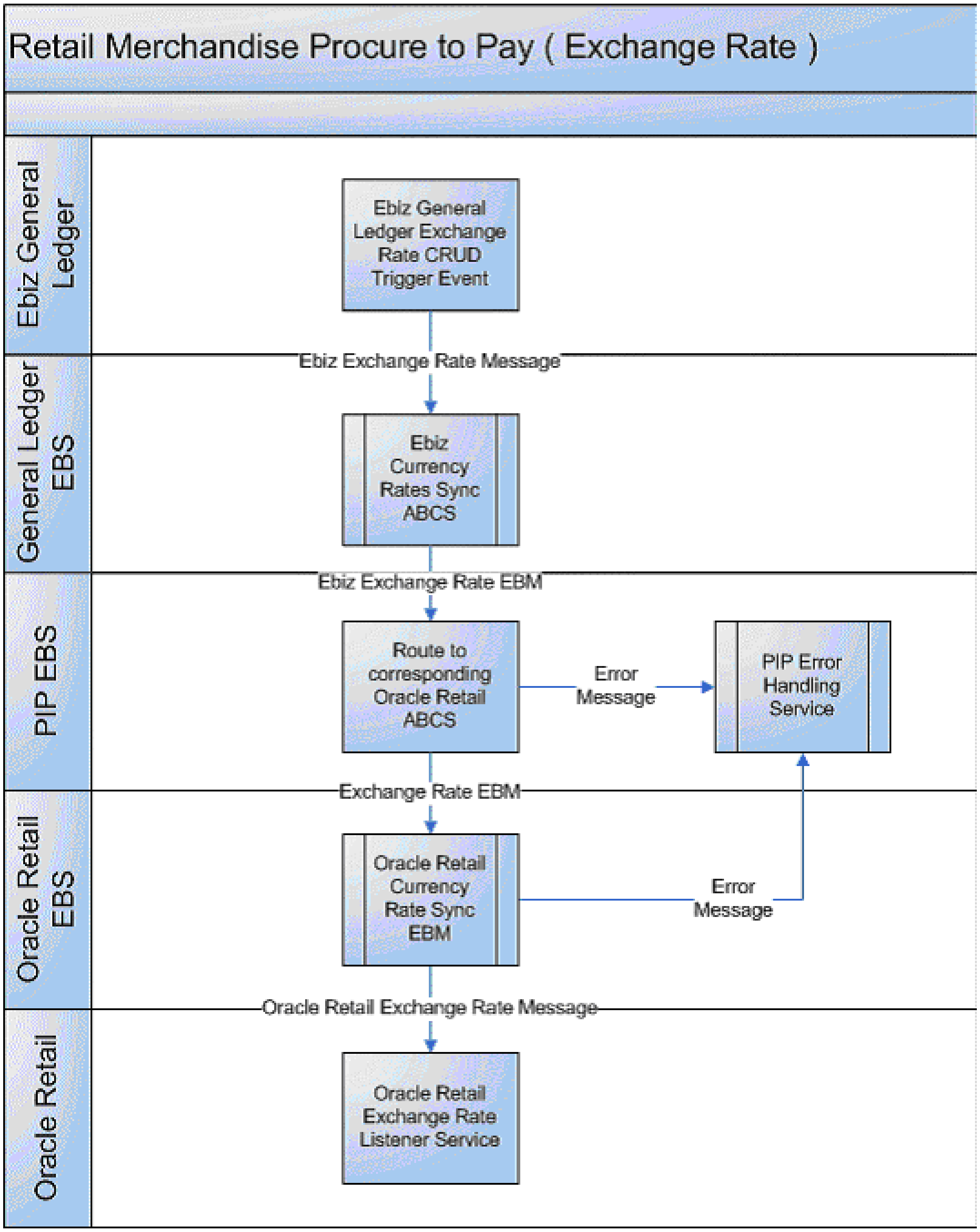
Currency codes and currency exchange rate types are manually maintained in both the Systems and mapped using DVM.

Solution Assumptions and Constraints

The integration design assumes that:

1. E-Business suite stores daily exchange rate for each day. For example, for the month of August 2009 for USD to Euro, there will be 31 entries with the specific conversion rates. Even if the same conversion rates apply to entire month, there will still be an individual exchange rate for each day.
2. The E-Business Suite application supports triangulation of currency exchange rates, but Oracle Retail does not.

3. Retail RIB Error Hospital holds all the Oracle Retail side errors and handles any notification on their side. Deleted currency exchange rates are not passed to Retail and the sync is one-direction only. This diagram illustrates the currency exchange rate integration flow:


Retail Merchandise Procure to Pay (Exchange Rate)
Ebiz General Ledger
General Ledger EBS
PIP EBS
Oracle Retail EBS
Oracle Retail
Ebiz General Ledger Exchange Rate CRUD Trigger Event
Ebiz Exchange Rate Message
Ebiz Currency Rates Sync ABCS
Ebiz Exchange Rate EBM
Route to corresponding Oracle Retail ABCS
Error Message
PIP Error Handling Service
Exchange Rate EBM
Oracle Retail Currency Rate Sync EBM
Error Message
Oracle Retail Exchange Rate Message
Oracle Retail Exchange Rate Listener Service

Currency Exchange Rate Integration Flow

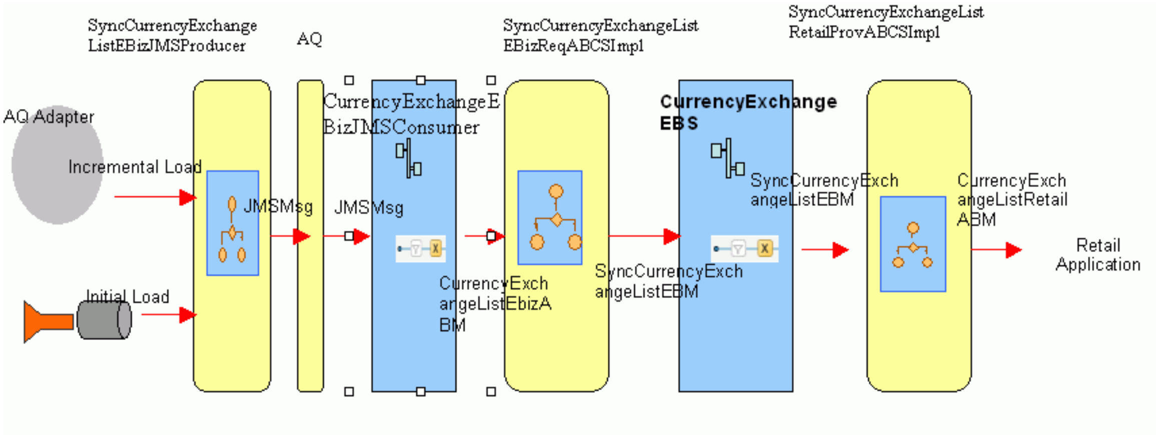
Currency Exchange Rate Integration Details

The integration flow uses these services:

- CurrencyExchangeListEbizJMSProducer
- CurrencyExchangeListEbizJMSConsumer
- CurrencyExchangeEBS
- SyncCurrencyExchangeListEbizAdapter
- SyncCurrencyExchangeListEbizReqABCSImpl

Sequence Diagram

This sequence diagram illustrates the currency exchange rate integration flow:



Currency exchange rate integration sequence diagram

Initial Loading of Currency Exchange Rates

This sequence diagram is applicable to initial load flows.

When you initiate the process:

- Oracle E-Business Suite invokes the CurrencyExchangeListEbizJMSProducer whenever a currency exchange rate is created or loaded into Oracle E-Business Suite.
- The CurrencyExchangeListEbizJMSProducer extracts all the currency exchange rates from the Oracle E-Business Suite database based on the dates provided in the Business Process Execution Language (BPEL) console and moves the currency exchange rates into the AIA_CURREXJMSQUEUE.
- The CurrencyExchangeListEbizJMSConsumer service picks up the message from the AIA_CURREXJMSQUEUE and invokes the SyncCurrencyExchangeListEbizReqABCSImpl, which transforms the message into the SyncCurrencyExchangeListEBM. Then, the SyncCurrencyExchangeListEBM invokes the CurrencyExchangeEBS.
- The CurrencyExchangeEBS receives the SyncCurrencyExchangeListEBM and invokes the SyncCurrencyExchangeListLogisticsProvABCSImpl.

- The SyncCurrencyExchangeListRetailProvABCSImpl receives the SyncCurrencyExchangeListEBM and transforms it into the CurrencyExchangeListRetailABM. Then, the Retail WebService is invoked with this transformed ABM. If the transaction is successful, then the cross reference values are populated for Retail columns for each of the Currency Exchange Id’s listed in the SyncCurrencyExchangeListEBM; otherwise it invokes the AIAAsyncErrorHandlingBPELProcess with an error.

Initial Loading of Currency Exchange Rates

The purpose of this flow is to load the existing currency exchange rates from E-Business Suite into Retail by triggering the CurrencyExchangeListEbizJMSProducer service.

Contact the Administrator to obtain the end point of the CurrencyExchangeListEbizJMSProducer web service.

How to start or run initial load of exchange rates

These are the steps to load the currency exchange rates initially from Oracle E-Business Suite to Retail.

1. Contact you’re SOA Administrator and obtain the endpoint location of the following process “CurrencyExchangeListEbizJMSProducer. The endpoint is like Obtain the endpoint URL for the following process “CurrencyExchangeListEbizJMSProducer”. The endpoint is like `http://<SOA_HOST>:<SOA_PORT>/orabpel/default/CurrencyExchangeListEbizJMSProducer/1.0`
2. Open the above endpoint and the following parameters appear,

from_currency		xsd:string
to_currency		xsd:string
from_date		xsd:date
to_date		xsd:date
conversion_rate_type		xsd:string
systemID		xsd:string

The “from_date” field is mandatory field. If the “to_date” is not specified, it takes the current date as the TO DATE. The valid date format is YYYY-MM-DD (2008-02-09). The rest of the elements are optional and can be used to limit the rates, which are required to be loaded from Oracle E-Business Suite to OTM. Use the appropriate date range based on the load of data and performance of the server.

If a USER defined conversion rate type needs to be loaded, provide the internal id / number of the rate type in the above parameter list.

Click on “Invoke” and the exchange rates within the input range specified are sent from Oracle E-Business into OTM.

Oracle RIB Interfaces

Inbound Interactions

- SyncCurrencyExchangeList (Asynchronous)
- Request Schema: CurrRateDesc.xsd

For more information, see the Oracle General Ledger User’s Guide.

Data Requirements

No data requirements exist for this process integration.

Core AIA Components

The currency exchange rate integration uses these components:

- CurrencyExchangeEBO
- CurrencyExchangeEBM
- CurrencyExchangeEBS

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 use a particular EBO or Enterprise Business Service (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 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 the process integration for currency exchange rate:

- CurrencyExchangeListEbizJMSProducer
- CurrencyExchangeListEbizJMSConsumer
- CurrencyExchangeEBS
- SyncCurrencyExchangeListEbizAdapter
- SyncCurrencyExchangeListEbizReqABCSImpl

You can 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

CurrencyExchangeListEbizJMSProducer

The CurrencyExchangeListEbizJMSProducer service is a Business Process Execution Language (BPEL) process. The Oracle system invokes this service when:

- A new currency exchange rate is created.
- An existing currency exchange rate is updated.
- A synchronization operation exists for a bulk load of currency exchange rates with SyncCurrencyExchangeListPSFTABM.

Oracle invokes the CurrencyExchangeListEbizJMSProducer service and reads all the Currency Exchange Rates from E-Business Suite DataBase based on the input values using the get_cur_conv_rates API whenever a currency exchange rate is created or updated.

This service populates JMSCorrelationID and puts the message in AIA_CURREXJMSQUEUE.

CurrencyExchangeListEbizJMSConsumer

CurrencyExchangeListEbizJMSConsumer is an Enterprise Service Bus (ESB) service. It has a JMS adapter called SyncCurrencyExchangeListEbizJMSConsumer. This adapter listens to the AIA_CURREXJMSQUEUE and picks up the messages for which JMSCorrelationID is SYNC. This invokes SyncCurrencyExchangeListOracleReqABCSImpl.

SyncCurrencyExchangeListEbizAdapter

The SyncCurrencyExchangeListEbizAdapter is an E-Business Suite process that listens to oracle.apps.gl.CurrencyConversionRates.dailyRate.specify business event. When any currency exchange rate is either created or updated, this event is raised. It extracts from_currency, to_currency, from_conversion_date, to_conversion_date, conversion_type from the business event. Then, CurrencyExchangeListEbizJMSProducer is invoked.

SyncCurrencyExchangeListEbizReqABCSImpl

The SyncCurrencyExchangeListEbizReqABCSImpl service is a BPEL process. This is a thin requester, which receives CurrencyExchangeEbizABM as input from CurrencyExchangeListEbizJMSConsumer. This message is transformed to SyncCurrencyExchangeListEBM, which populates EBM Header and the cross-reference table and invokes CurrencyExchangeEBS

CurrencyExchangeEBS

CurrencyExchangeEBS routes all currency exchange related actions, such as create and update of currency exchange rate as Sync Currency Exchange Rate message to SyncCurrencyExchangeLogisticsProvABCSImpl or Composite Application Validation System (CAVS) based on the filter condition and operations. For this particular integration, only Sync action is available. Updates and creates are done using the Sync action.

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”

Suppliers Information Integration

This section provides an overview of the process integration for initial loading and incremental synchronization of suppliers' information between Oracle Payables and Oracle Retail Merchandising System (RMS) and discusses:

- Supplier integration details
- Oracle Payables interfaces
- RMS interfaces
- Data requirements
- Core AIA components
- Integration services

Supplier Integration Overview

In the integrated environment, Oracle Payables acts as a payable, and RMS handles supplier payments, merchandise write-offs, and prepaid adjustments.

Merchandise suppliers are suppliers of goods and services that the retailer sells to customers. Oracle Payables and RMS require sharing of suppliers' information between them. RMS requires the supplier information for several key functions including creation and management of items and purchase orders. Oracle Payables requires suppliers' information for supplier payment. For end-to-end business integration, same supplier instance and related information must be shared between these two systems.

Between the two systems, Oracle Payables is the source of valid suppliers (vendors in Oracle Payables) and their Remit to Location and Order from addresses

The supplier integration synchronizes supplier's information from Oracle Payables to RMS through these integration flows:

- Load initial suppliers from Oracle Payables to RMS: Enables the loading of all active merchandise suppliers, the current effective supplier locations and their current effective remit and order to addresses.
- Incremental creation and updates of suppliers from Oracle Payables to RMS: Enables the synchronization of incremental creation and updates of the active suppliers from Oracle Payables to RMS.

Note: For incremental loads, Oracle Payables doesn't propogate deleted suppliers or inactive date information to Retail.

This integration is not a point-to-point integration between *Oracle* Payables and RMS. An AIA layer serves as an intermediate thin layer of application between *Oracle* Payables and RMS. As a part of the supplier integration, *Oracle* Payables sends the suppliers' information to the AIA layer and the AIA layer delivers the information to RMS. The AIA layer performs message filtering, message transformation, and message routing. Because this integration is not a point-to-point integration, the vendor number (ID) in Oracle is not similar to the supplier number (ID) in Oracle Retail.

Prerequisites

These are the prerequisites for this integration:

For more information about initial load of the ORGANIZATION_ID and payment term cross reference table, see 'Implementing Oracle Retail Merchandise Integration Pack for Oracle Financials. See section Setting up Cross-Reference for Oracle Retail IDS and Oracle Entities.

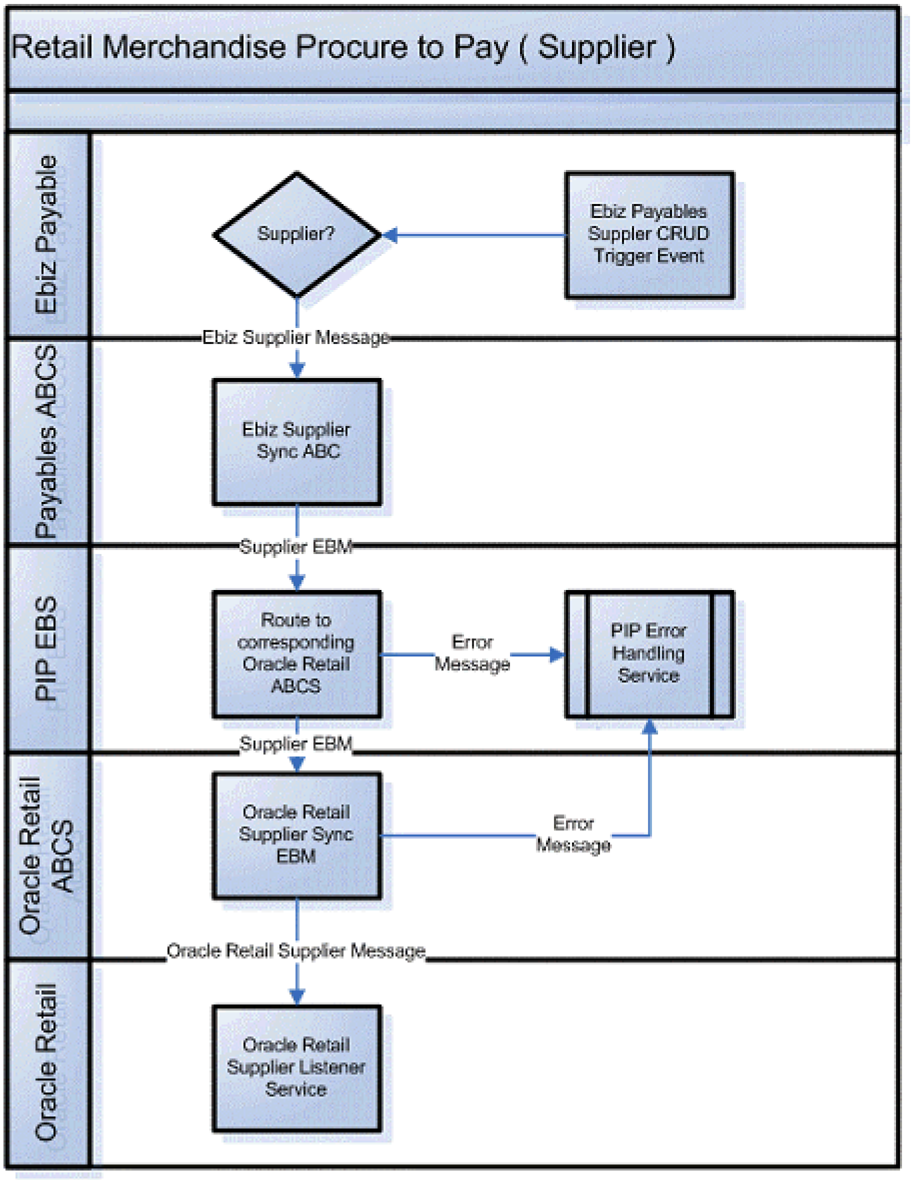
- Payment terms and Freight terms synchronization using manual process. Must be manually synced in both systems.

Solution Assumptions and Constraints

The integration design assumes that:

1. Oracle Payables is the source system for merchandise suppliers, their contacts, locations, addresses and other attributes
2. You can create suppliers and suppliers' locations in Oracle Payables.
3. You can maintain the relationship between suppliers, suppliers' locations in Oracle Payables. This integration is a one-way synchronization. Any update to supplier information in RMS is not synchronized with Oracle Payables. The volume of payload (SupplierPartyList) that is handled by the process depends on the server configuration.
4. Oracle sends the suppliers in batches based on different criteria.
5. The Oracle system sends all the related information over to AIA. For example, if an address is changed, then the supplier linked to that address is sent.

This diagram illustrates the supplier integration flow:


Retail Merchandise Procure to Pay (Supplier)
Ebiz Payable
Payables ABCS
PIP EBS
Oracle Retail ABCS
Oracle Retail
Supplier?
Ebiz Payables Suppler CRUD Trigger Event
Ebiz Supplier Message
Ebiz Supplier Sync ABC
Supplier EBM
Route to corresponding Oracle Retail ABCS
Error Message
PIP Error Handling Service
Supplier EBM
Oracle Retail Supplier Sync EBM
Error Message
Oracle Retail Supplier Message
Oracle Retail Supplier Listener Service

Supplier integration flow

Supplier Integration Details

These services are common sync supplier integration flows:

- SupplierPartyListEBizJMSProducer
- SupplierPartyListEBizJMSConsumer
- SupplierPartyEBS

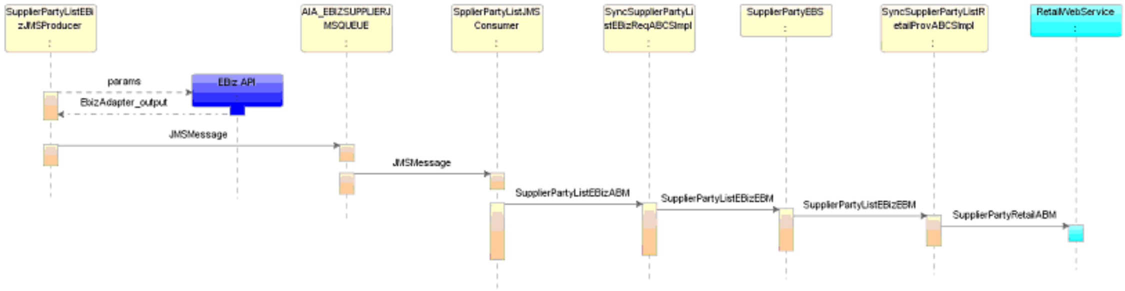
These services are specific to sync supplier integration flow:

- SyncSupplierPartyListEbizReqABCSImpl

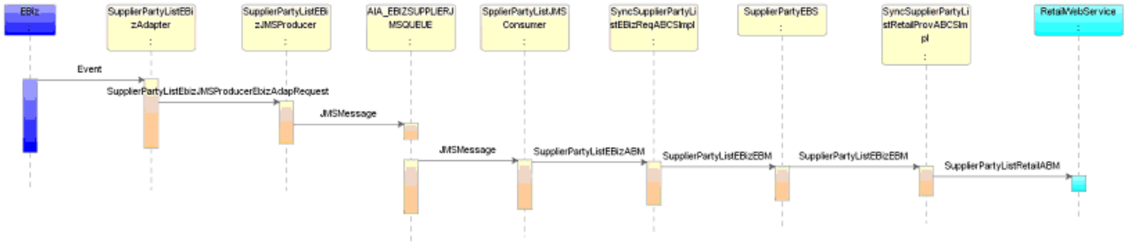
- SyncSupplierPartyListRetailProvABCSImpl

Sequence Diagram

This sequence diagram illustrates the supplier information integration flow:



Supplier Sequence Initial Load


EBiz
SupplierPartyListEBizAdapter
SupplierPartyListEBizJMSProducer
AIA_EBIZSUPPLIERJMSQUEUE
SpplierPartyListJMSConsumer
SyncSupplierPartyListEBizReqABCSImpl
SupplierPartyEBS
SyncSupplierPartyListRetailProvABCSImpl
RetailWebService
Event
SupplierPartyListEbizJMSProducerEbizAdapRequest
JMSMessage
JMSMessage
SupplierPartyListEBizABM
SupplierPartyListEBizEBM
SupplierPartyListEBizEBM
SupplierPartyListRetailABM

Supplier Sequence Incremental Load

This sequence diagram is applicable to initial and incremental load and sync integration flows.

When you initiate the process:

1. Oracle Payables invokes the SupplierPartyListEBizJMSProducer service with SyncSupplierPartyListEbizABM.

   This service puts this ABM in the AIA_EBIZSUPPJMSQUEUE with JMSCorrelationID ID set to Sync/Create/Update.

2. The SupplierPartyListEbizJMSConsumer service listens on the AIA_EBIZSUPPJMSQUEUE and invokes SyncSupplierPartyListEbizReqABCSImpl in case of SyncSupplierPartyListEbizABM.

3. The SyncSupplierPartyListEbizReqABCSImpl transforms the SupplierPartyListABM message into SupplierPartyListEBM.

   Transformation does cross-referencing for system-specific values and calls the SupplierPartyEBS with operation SyncSupplierPartList. SupplierPartyEBS is a routing ESB service with several operations on the supplier EBO.

4. SupplierPartyEBS routes the SupplierPartyListEBM to Retail provider ABCS implementation. It also provides CAVS routing. It is the Enterprise Business Service to route all SupplierParty related actions like SyncSupplierPartyList. This process is used for Ebiz-Fleet integration on AIA 2.5.

5. SyncSupplierPartyListRetailProvABCSImpl transforms SupplierPartyListEBM to Retail supplier message ABM.

Transformation applies the DVM and invokes create or update web service from RMS. It also updates the cross-reference table after the Retail web service call.

RMS

Inbound Interactions:

- SupplierServicePortFactory
    - Operation: createSupplierCollectionDesc, Request Schema: SupplierCollectionDesc.xsd, Response Schema: SupplierCollectionRef.xsd
    - Operation: updateSupplierCollectionDesc, Request Schema: SupplierCollectionDesc.xsd, Response Schema: SupplierCollectionRef.xsd
    - Operation: createSupplierDesc, Request Schema: SupplierDesc.xsd, Response Schema: SupplierRef.xsd
    - Operation: updateSupplierDesc, Request Schema: SupplierDesc.xsd, Response Schema: SupplierRef.xsd

For more information, see the *RMS Operations Guide*, Volume 2, “Web Services Overview”

Data Requirements

The process integration for supplier information requires these data:

- Suppliers must be assigned to at least one GL business unit (set of books)
- For this integration, Oracle suppliers must be created with These status attributes:
    - Classification of supplier. Other type of vendors such as attorneys, employees, and HCM are not synchronized with Oracle Retail. All suppliers except employee are synced.
    - The Open for ordering option is selected. If the supplier is approved but is not open for ordering, the AIA layer on the RMS side changes its status to inactive. Ebiz doesn’t have an approval status.

Note: Only contacts associated with supplier sites or addresses will be synchronized from Oracle Payables to RMS.

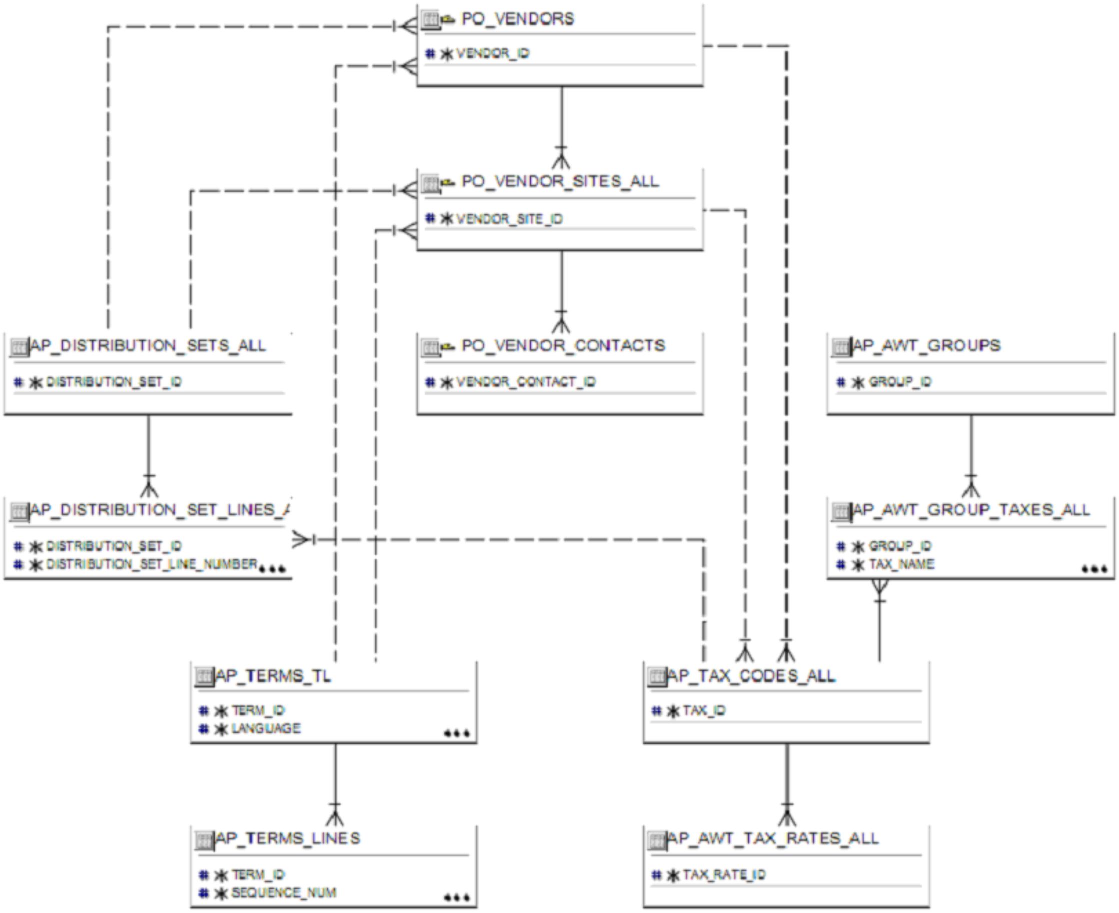
For Retail to Ebiz integration, only certain suppliers with specific criteria are sent to Retail:

- Supplier must have at least one or more address or site
- Supplier address or site must be a purchasing and/or a payment purpose.
- If a supplier has one address or site, it must be both a purchasing and payment purpose.
- If a supplier has two or more addresses, one address may be a purchasing site and other address a payment site.

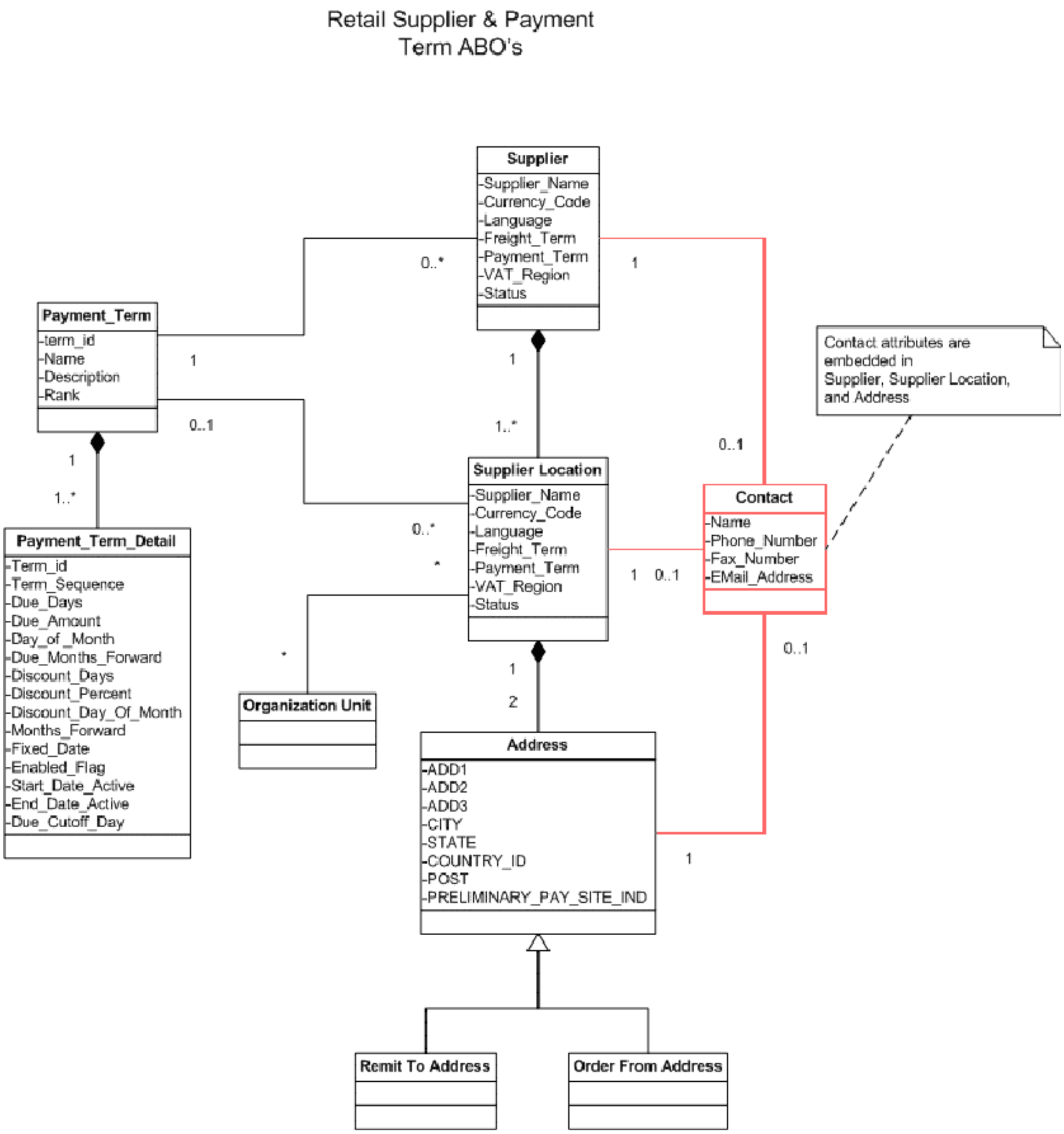
- Supplier address or site without a purchasing or payment purpose will not be sent to Retail.
- In the contact details and purpose, the phone area code, fax number and email address are meant for general contact information such as a main operator number or a general company inquiry email address. This information should not be passed to Retail.
- Under Company Profile --> Contact directory, there is contact information for specific people and their specific contact information. For example, the Account Receivable person's phone #, name, and email address is listed here. This information is passed to Retail.
- Supplier Recommendation
    - Data Cleansing prior to Supplier Initial Load: It is recommended that for each E-Business Suite supplier, you must select a supplier site as a primary pay site. The values on the primary pay site are used as the default values for any supplier attributes such as payment terms and freight terms that are missing during the integration.
    - Integration Supplier Logic: For a supplier that does not have a payment or freight terms, the primary pay site is used for the default values. If the supplier does not have a primary pay site, then the first purchasing site is used. If no payment or freight terms are defined at supplier or any supplier purchasing site, and there is no primary pay site, then the supplier transfer fails or errors.

Class Diagram

The following diagrams show supplier relation in E-Business Suite and Retail:



Supplier Relations in E-Business Suite



Supplier Relation in Retail

Core AIA Components

The supplier Information integration uses these components:

- Merchandise Supplier EBO (SupplierPartyEBO)
- Merchandise Supplier EBS (SupplierPartyEBS)
- Merchandise Supplier EBM (SupplierPartyEBM)

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 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 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 the process integration for supplier information:

- SupplierPartyEbizJMSProducer
- SupplierPartyEbizJMSConsumer
- SyncSupplierPartyListEbizReqABCSImpl
- SupplierPartyEBS
- SyncSupplierPartyListRetailProvABCSImpl

You can 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

SupplierPartyEbizJMSProducer

Oracle Payables invokes the SupplierPartyEbizJMSProducer service for the initial load and incremental create and update operation. The same service can support the bulk update of suppliers from Oracle to RMS.

This is a BPEL process. This process receives different types of message in CreateSupplierParty, UpdateSupplierParty, and SyncSupplierPartyList with three different operations. It publishes the message in the queue with the help of JMSAdapter. JMSHeader is set before enqueue of the message. Values for the JMSHeader correlation are Sync, Create, and Update. The message consumers pick up the message based on the header value.

SupplierPartyListEbizJMSConsumer

SupplierPartyListEbizJMSConsumer is an ESB. It has three JMSAdapter services for dequeue of the message from the queue. After dequeing the messages, it calls for the respective OracleReqABCSImpl (Create, Update, or Sync).

This service has these JMS adapters:

- SyncSupplierPartyListEbizJMSConsumer

SyncSupplierPartyListEbizReqABCSImpl

SupplierPartyListEbizJMSConsumer invokes the SyncSupplierPartyListEbizReqABCSImpl service for the list of existing suppliers during the initial integration with other applications.

This is a single operation service that has SupplierPartyEBS as a partner service. This accepts a Oracle SupplierPartyListEBM message as a request and does not return a response to the calling service. This service is a BPEL process.

This service performs these actions:

- Accepts a SyncSupplierPartyListReqMsg message from Oracle. This message contains an XREF for suppliers and suppliers' addresses.
- Transforms SyncSupplierPartyListReqMsg to SyncSupplierPartyListEBM. While it is transforming ABM to EBM, cross-references are looked up for SUPPLIERPARTY_ID, SUPPLIERPARTY_LOCATION_ID, and SUPPLIERPARTY_ADDRESS_ID.
- Sends SyncSupplierPartyListEBM messages as an input to the SyncSupplierPartyList operation in the SupplierPartyEBS service.

SupplierPartyEBS

SupplierPartyEBS performs all the supplier-related actions such as Create Supplier, Update Supplier, and Sync Supplier. It routes the incoming Sync, Update, or Create messages to the respective ProviderABCSImpl. No transformation is done in this EBS.

SupplierPartyEBS

These operations on the SupplierPartyEBS are used in the supplier information integration flow:

- SyncSupplierPartyList

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"

SyncSupplierPartyListRetailProvABCSImpl

SyncSupplierPartyListRetailProvABCSImpl

SyncSupplierPartyListRetailProvABCSImpl is a single operation service. This service is a BPEL process.

This process performs these actions:

- Receives input from SupplierPartyEBS.
- Accepts a SyncSupplierPartyListEBM message. A SyncSupplierPartyListEBM message contains these details:
    - Standard supplier attributes
    - Supplier contacts information
    - Supplier location information
    - Supplier location contact information
    - Supplier location remit address information
    - Supplier location remit address contact information
    - Supplier location order address information
    - Supplier location order address contact information
    - Supplier location business unit
- Transforms SyncSupplierPartyListEBM into a SyncSupplierRetailABM message.

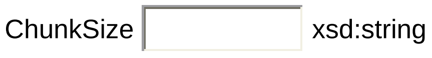
  During transformation, the DVM lookups change the value for language code, state, country code, currency code, SupplierParty address type, and SupplierParty status code.
- Invokes the Create/UpdateSupplierPartyList WSDL after transforming the message from the SyncSupplierEBS to SyncSupplierPartyList ABM and wrapping the ABM in WSDL.
- Updates the cross-references for Oracle Retail after the Oracle Retail web service call.

How to start or run initial load of Supplier Parties

1. Obtain the Endpoint URL for the InitialLoadSupplierPartyListEbizAdapter. The endpoint is as below

http://<SOA_HOST>:<SOA_PORT>/orabpel/default/InitialLoadSupplierPartyListEbizAdapter/1.0.

2. On clicking on the above link, the following parameter displays


ChunkSize xsd:string

3. Enter an appropriate value for the chunk size field based on the memory of the system, the number of suppliers present in the Oracle E-Business Suite for initial load. It is appropriate to give this value between 50 and 100 or even smaller based on the kind of supplier profiles existing in the system.
4. Click on the “Invoke” button. This starts the initial load of the suppliers from E-Business to Retail.

Performance tuning parameters:

In case of any issues during the initial load, the below properties can be fine-tuned by referring to the SOA Guide.

Go to the Fusion Middleware machine and change the domain level configurations as given below:

1. <SOA_ORACLE_HOME>/bpel/domains/<domain_name>/config/domain.xml
   Property Name: syncMaxWaitTime
   Property Value: 120

2. <SOA_ORACLE_HOME>/integration/esb/config/esb_config.ini
   Property Name: xa_timeout
   Property Value: 120
   Property Name: jms_receive_timeout
   Property Value: 120

3. <SOA_ORACLE_HOME>/j2ee/<domain_name>/application-deployments/orabpel/ejb_ob_engine/orion-ejb-jar.xml
   Property Name: xa_timeout
   Property Value: 120

4. <SOA_ORACLE_HOME>/j2ee/<domain_name>/config/transaction-manager.xml
   Property Name: transaction-timeout
   Property Value:>120

5. In BPEL Console Go to Configuration Tab and change the value for "auditLevel" to off/minimal

6. In opmn.xml,process type id="oc4j_soa" at start-parameters give as -Xmx2048M

7. Provide a lesser value for the chunk size to run the initial load.

Alternative Option:

An alternative approach to running the initial load of suppliers from Oracle E-Business into Retail is described below.

The initial load of supplier information from Oracle E-Business to Retail can also be done in the following manner provided the range of supplier ids are known:

1. Obtain the Endpoint location for the SupplierPartyListEbizJMSProducer. It is like the below URL
   http://<SOA_HOST>:<SOA_PORT>/orabpel/default/SupplierPartyListEbizJMSProducer/1.0

2. Choose the operation as ProduceSupplierPartyABMInitLoad.

3. Enter the data in From Vendor ID and To Vendor ID fields for which the data should be synchronized from Oracle E-Business Suite to Retail.

4. Click “Invoke” – the suppliers present in Oracle E-Business Suite between the “from” and “to” values are synchronized to Retail.

Chapter 3: Chart of Accounts Validation Integration Flow

This chapter describes how to request the chart of accounts combination validation and discusses:

- Oracle Retail interfaces
- Oracle GL interfaces
- Data requirements
- Core Application Integration Architecture (AIA) components
- Integration services

Requesting Chart of Accounts Combination Validation

Oracle General Ledger (GL) is the system of record for chart of accounts segment combinations. Chart of accounts (segments) combinations are set up manually in Oracle Retail Merchandising System (RMS), Retail Invoice Matching (ReIM), and Retail Sales Audit (ReSA). This manual setup enables assignment of transaction data in sales audit, stock ledger, and invoice match to specific account codes. ReIM also creates segment combinations dynamically during the invoice match transaction processing.

While creating the valid segment combinations, Oracle Retail validates each created combination individually against Oracle GL. Any valid segment combination in Oracle Retail must exist as a valid combination in Oracle GL. Oracle Retail publishes the segment combination to Oracle GL for validation. Oracle GL verifies the combination and returns the status to Oracle Retail. If the combination is valid, the combination is stored in the Oracle retail database tables for future validation purpose.

Oracle Retail sends these data for validation to Oracle GL:

- Requesting system (RMS, ReIM, or ReSA)
- Set of books (GL business unit)
- Segment or ChartField combination values

Oracle GL sends the validation status whether the combination is valid. The AIA layer copies all the other information from the original request and sends these response data to Oracle Retail:

- Requestion system (RSM, ReIM, or ReSA)
- Set of books (GL business unit)
- Date
- Validation status – valid or invalid

Solution Assumptions and Constraints

This design assumes that:

- The Oracle Retail request application business message (ABM) contains a field called requesting_system that identifies the system that is requesting the service so that the correct Retail segment is retrieved from the Domain Value Map (DVM) lookup.

  DVM values (segments to Oracle accounts) may be set up differently for each system (RMS, ReSA, or ReIM).

- Oracle GL accepts all the AIA Common values in the provider ABM.
- Therefore, no DVM lookups are required on the Oracle provider side. The DVM values are entered in AIA and E-Business Suite. Oracle GL receives the array of the values for each segment. If the structure is segement1-segment2-segment3-segementn, then Oracle GL expects that the array size (in this case is n) and the array of value in the order of 1 to n is passed.

  Multiple sets of GL accounts can be sent from Oracle Retail to the E-Business Suite validation web service.

- Once the validated chart of account is stored in Retail, Oracle GL doesn’t send any updates if chart of account becomes invalid.

Chart of Accounts Combination Validation Integration Flow

These services are delivered with the chart of accounts combination validation integration flow:

- ProcessGLAccountValidationRetailReqABCSImpl
- ChartOfAccountsEBS
- ProcessGLAccountValidationEbizProvABCSImpl

This diagram illustrates the chart of accounts combination validation integration flow:

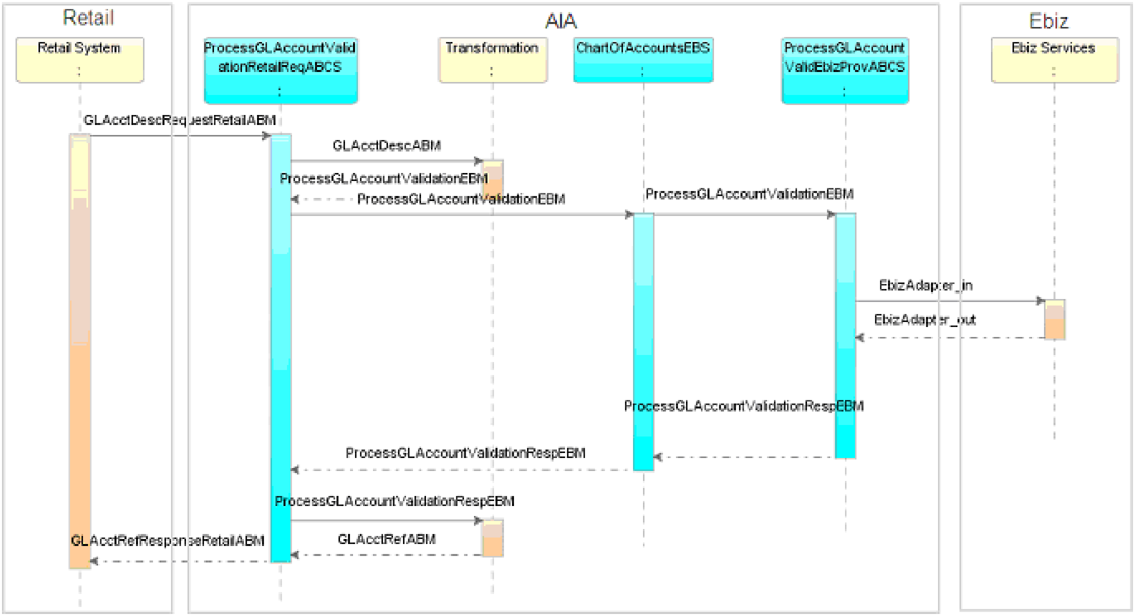

Retail
Retail System
AIA
ProcessGLAccountValidationRetailReqABCS
Transformation
ChartOfAccountsEBS
ProcessGLAccountValidEbizProvABCS
Ebiz
Ebiz Services
GLAcctDescRequestRetailABM
GLAcctDescABM
ProcessGLAccountValidationEBM
ProcessGLAccountValidationEBM
ProcessGLAccountValidationEBM
EbizAdapter_in
EbizAdapter_out
ProcessGLAccountValidationRespEBM
ProcessGLAccountValidationRespEBM
ProcessGLAccountValidationRespEBM
GLAcctRefResponseRetailABM
GLAcctRefABM

Chart of accounts combination validation sequence diagram

When the chart of accounts validation process is initiated:

1. Oracle Retail calls the ProcessGLAccountValidationRetailReqABCSImpl service with the ProcessGLAccountValidationRetailABM (GLAcctDesc.xsd) whenever a GL account is required to be validated.

   The ProcessGLAccountValidationRetailReqABCSImpl service transforms the ProcessGLAccountValidationRetailABM into the ProcessGLAccountValidationEBM and then invokes the ChartOfAccountsEBS. When it returns from the ChartOfAccountsEBS, it transforms the ProcessGLAccountValidationResponseEBM to the ProcessGLAccountValidationResponseRetailABM (GLAcctRef.xsd) and sends the response back to Oracle Retail.

2. ChartOfAccountsEBS invokes ProcessGLAccountValidationListEbizProvABCSImpl or Composite Application Validation System (CAVS) based on the routing rule.

3. ProcessGLAccountValidationListEbizProvABCSImpl invokes the Oracle GL_CHARTOFACCOUNT service with ProcessGLAccountValidationEBM for validating the GL account.

   Then ProcessGLAccountValidationOracleProvABCSImpl returns ProcessGLAccountValidationResponseEBM to ChartOfAccountsEBS.

Oracle Retail Interfaces

Retail Outbound Interactions:

- Name: ProcessGLAccountValidation: Oracle Retail (RMS, RESA, or REIM) invokes this service to determine whether an account is valid with Oracle GL.

- Interaction Pattern: Synchronous Request - Response
- Request Schema: GLAcctCollectionDesc.xsd, GLAcctDesc.xsd
- Response Schema: GLAcctCollectionRef.xsd, GLAcctRef.xsd.

Data Requirements

The segment combinations in the GL ACCOUNT MAINTENANCE (Sales Audit), GL CROSS REFERENCE (RMS), and GL CROSS REFERENCE (ReIM) forms must be entered manually.

Core AIA Components

The chart of accounts combination validation integration flow uses these components:

- ChartOfAccountsEBO
- ChartOfAccountsEBM
- ChartOfAccountsEBS

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 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 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

The chart of accounts combination validation integration flow uses these services:

- ProcessGLAccountValidationRetailReqABCSImpl
- ChartOfAccountsEBS
- ProcessGLAccountValidationEbizProvABCSImpl

You can use the Integration Scenario Summary page in the Oracle AIA Console to search for and view integration scenarios that use a particular Applications Business Connector Service (ABCS).

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

ProcessGLAccountValidationRetailReqABCSImpl

Oracle Retail (RMS, ReSA, or ReIM) invokes this requestor ABCS to validate a GL account. This service takes ProcessGLAccountValidationRetailABM (GLAcctDesc.xsd) to validate a GL account. It transforms ProcessGLAccountValidationRetailABM into the ProcessGLAccountValidationEBM and then invokes ChartOfAccountsEBS. When ChartOfAccountsEBS returns with ProcessGLAccountValidationResponseEBM, it is then transformed into ProcessGLAccountValidationResponseRetailABM (GLAcctRef.xsd) and sends it to the calling application.

In this requestor ABCS, the Set of Books ID is mapped to the ChartOfAccountIdentification field in the EBM. During this transformation, the Set of Books ID uses the BUSINESS_UNIT DVM lookup to transform the Set of Books ID from Oracle Retail and retrieves the common value and loads it into the ChartOfAccountIdentification field in the EBM.

When ProcessGLAccountValidationResponseEBM returns regardless of whether the account is valid, a transformation occurs to populate the GLAcctRef account_status field. A CHARTOFACCOUNTS_ACCOUNTSTATUS DVM table is used to transform the AIA Common/PSFT value to the Oracle Retail value.

ProcessGLAccountValidationRetailReqABCSImplExt is a partner link service. This is an extensibility service that takes ProcessGLAccountValidationRetailABM, ProcessGLAccountValidationRetailABM, ProcessGLAccountValidationEBM, and ProcessGLAccountValidationResponseEBM as inputs and returns the same. This service enables the customer to filter, augment, and validate the input or output.

ChartOfAccountsEBS

ChartOfAccountsEBS calls ProcessGLAccountValidationRetailReqABCSImpl. It expects the ProcessGLAccountValidationEBM encapsulating the ChartOfAccountsEBO and sends the ProcessGLAccountValidationEBM to the Oracle ChartField validation service. In turn, it returns whether the account is valid to the Retail ProcessGLAccountValidationRetailReqABCSImpl 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.”

ProcessGLAccountValidationEbizProvABCSImpl

The ProcessGLAccountValidationOracleProvABCSImpl service receives ProcessGLAccountValidationEBM as an input from ChartOfAccountsEBS. This is a single operation service. This service expects ProcessGLAccountValidationEBM and publishes ProcessGLAccountValidationEBM to Ebiz Code Combination Validate API. In turn, it returns a notification of validation results to ChartOfAccountsEBS.

ChartOfAccountsEBS invokes ProcessGLAccountValidationEbizProvABCSImpl with ProcessGLAccountValidationEBM. EBM_Header variable is assigned with the header information of ProcessGLAccountValidationEBM.

ProcessGLAccountValidationEbizProvABCSImplExt's PreProcessEBM is invoked with ProcessGLAccountValidationEBM based on a parameter in AIAConfig file, which is for custom extensibility.

There is a loop to handle the validation of Chart of Accounts, before the loop, it maps the request EBM to the response EBM. It get the chart of account one by one from the EBM, and assign the segments in the COA to the input parameter of Ebiz Adapter, then calls the validation, assigns the valid flag based on the return value. If the account is valid, then sets return valid CCID to the response variable; otherwise, set the received CCID to the response variable, appends the response variable to the response variable list.

There is a transformation, which transforms the valid indicator and the CCID in response variable list to ProcessGLAccountValidationResponseEBM. This transformation is just an assignment because the response variable list type is same with EBM. No DVM lookups are needed.

ProcessGLAccountValidationEbizProvABCSImplExt's PostProcessEBM is invoked with ProcessGLAccountValidationResponseEBM based on a parameter in AIAConfig file, which is for custom extensibility.

ProcessGLAccountValidationEbizProvABCSImpl returns the ProcessGLAccountValidationResponseEBM to the caller ChartOfAccountsEBS.

Chapter 4: Process Integration for Inventory Valuation and Revenue Recognition Accounting Entries

This chapter provides an overview of the process integration for accounting entries from Oracle Retail Sales Audit (ReSA), Retail Merchandising System (RMS), and Retail Invoice Matching (ReIM) to Oracle General Ledger (GL) and discusses how to:

- Perform setup tasks
- Configure and generate data in Oracle Retail applications
- Pick up and transform the data
- Configure and run the process integration for accounting entries

Process Integration for Inventory Valuation and Revenue Recognition Accounting Entries Overview

The process integration for accounting entries enables you to record the financial impact of changes to sellable store and warehouse inventory. It also records the financial impact of sales and returns, cash reconciliation, and void transactions from stores.

The system sends sales audit and stock ledger data in Oracle Retail to Oracle GL through accounting entries. Similarly the system sends the prepaid invoice reversals and write-offs of aged receipts Oracle GL. RMS, ReIM, and ReSA are the source for accounting entry. After successful completion of data transfer, the system deletes the data from the source interface tables.

This integration uses the Oracle Data Integrator (ODI) application to transfer the data from the Oracle Retail applications to the Oracle GL application.

The process integration between Oracle Retail (RMS, ReSA, and ReIM) and Oracle GL supports these integration flows:

- Post stock ledger from RMS to Oracle GL.
- Post channel sales, cash, and deposits from ReSA to Oracle GL.
- Post write-offs (aged receipts, not invoiced to ledger) from RMS to Oracle GL.
- Post prepaid invoice reversals after receipt from ReIM to Oracle GL.

Business Process Flow for the Integration of Accounting Entries

The overall process includes:

Oracle Retail	**Step1**: Prepare the interface data
Integration Process (ODI)	**Step2**: Integration process will wait for the data to arrive in Retail staging tables. **Step3**: Load and Transform retail data into E-Business Suite GL interface table. **Step4**: Delete the data from the source staging tables.
E-Business Suite Financials	**Step5**: Launch journal import concurrent program to import data from GL interface table to journal transaction tables. An event will be raised after the concurrent program is completed. Then manually post to GL.
Integration Process (AIA/BPEL)	**Step6**: Update the AIA XREF table using a BPEL service that listens to the event raised in step5.

Prerequisites

Before performing this process integration, ensure that:

1. These Domain Value Maps (DVMs) have been entered on the Enterprise Service Bus (ESB) console, and exported, also set up in the Oracle system:
    a. BUSINESS_UNIT
    b. CURRENCY_CODE
    c. PAYABLE INVOICE TAX CODE
        - The currency exchange rate is synchronized between Oracle Retail and E-Business Suite Financials.
        - The journal category and source are set up in E-Business Suite Financials.

The AIAConfigurationProperties.XML file is set up.

Facts and Constraints

Oracle Retail:

1. RMS, ReSA, and ReIM are the source of the accounting entry flow.
2. Oracle Retail populates one of the staging tables: STG_FIF_GL_DATA or IM_FINANCIALS_STAGE (depending on whether the RMS or ReIM package is running) on a scheduled basis.
3. Data in the Oracle Retail staging tables is deleted after the ODI job completes successfully.

Oracle GL:

- Oracle GL has one table to receive data; GL_INTERFACE

Performing Setup Tasks

Setup tasks specific to E-Business Suite Financials (General Ledger)

- Apply patch 7261105:R12.GL.B
- Enable event oracle.apps.gl.Journals.journal.completeImport if customer wants the AIA XREF to be populated
- Synchronize currency exchange rate between Oracle Retail and E-Business Suite.
- Set up journal category and source.
- The following DVMs have been entered on the ESB console:
    - CURRENCY CODE
    - BUSINESS UNIT
    - PAYABLE INVOICE TAX CODE
- The following AIA XREF has been populated
    - ORGANIZATION_ID

Setup tasks specific to this integration layer

You must complete these tasks in the integration layer:

- EXPORT DVMS

  During install process, the Oracle Universal Installer will prompt for a location to export the DVMs. Use that location to export the following DVMs from ESB:
- CURRENCY CODE
- BUSINESS UNIT
- PAYABLE INVOICE TAX CODE

  Set up the customer email address and data pooling time in AIA Configuration properties XML file. This file exist on <AIA_HOME>/Infrastructure/SeedData. We'll add a ModuleConfiguration element to this file, RetailToE-BusinessSuiteAccountingEntry for GL entry. See appendix sample data for more.

Configuring and Generating Data in Oracle Retail

This section discusses:

- Configuring the data in Oracle Retail

- Generating the data in Oracle Retail

Configuring the Data in Oracle Retail

Before running any accounting data, perform these tasks in RMS and ReIM:

- Set up general ledger (GL) account cross-reference
- Define GL options.

For more information, see the *RMS User Guide,* “Financial Management,” “Stock Ledger,” “Maintain general ledger cross reference” and the *ReIM User Guide,* “System Administration,” ”General Ledger Accounts”

Generating the Data in Oracle Retail

RMS stages GL data for subsequent upload into the integrated financial system. A set of batch processes gather and organize the data before using it to populate the related staging table.

These batch designs are included in this functional area

- FIFGLDN1.PC
- FIFGLDN2.PC
- FIFGLDN3.PC

For more information, see the *RMS Operations Guide.*

In ReIM, the batch process engages in these high-level steps:

1. Performs any resolution actions (for example, initiate the creation of payment documents).
2. Calls the posting process to write applicable financial accounting transactions to the financials staging table, IM_FINANCIALS_STAGE.

The processing occurs after discrepancies for documents have been resolved by resolution documents. Once all of the resolution documents for a matched invoice are built, and all of the RCA/RUA external processing has been confirmed, the process inserts financial accounting transactions to the financials staging table, to represent the resolution and consequent posting of the invoice. The process also inserts financial accounting transactions for the approved documents that are being handled.

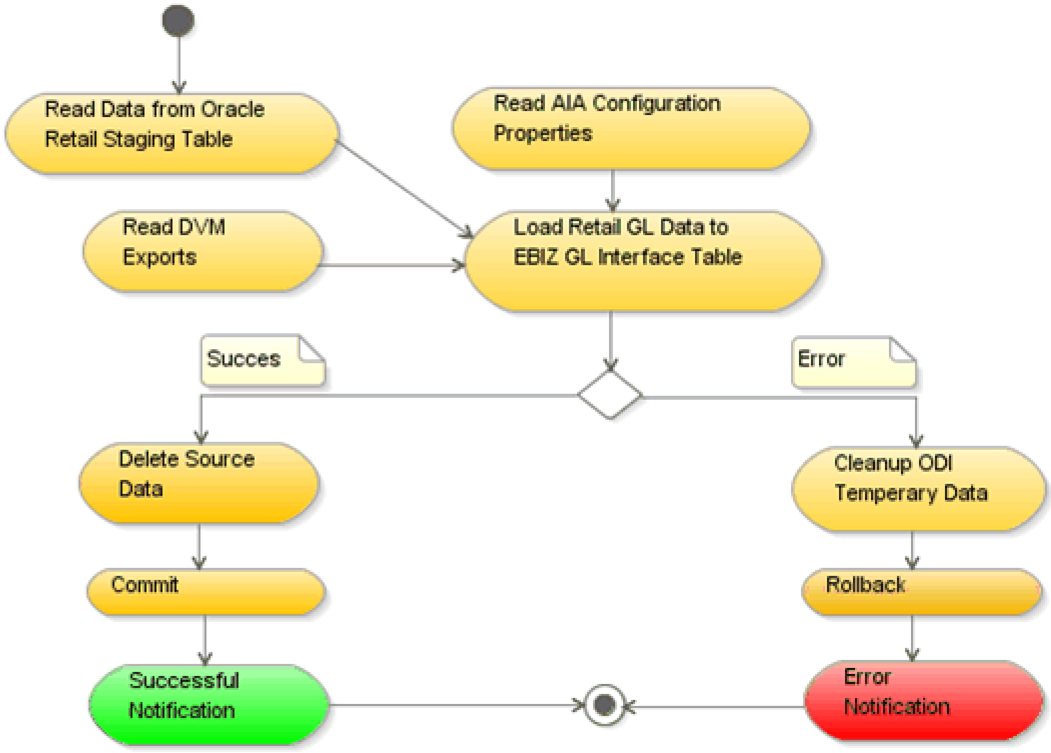
Once all of the transactions have been written, the process switches the status of the current invoices or documents to “Posted”, and then moves on to the next invoice or document. If a segment look-up fails, the failed record is written to a financials error table.

For more information, see the *ReIM Operations Guide.*

Picking Up and Transforming the Data

The process integration for accounting entries uses ODI to pick up the data from RMS, ReSA, and ReIM, transform it, and load it into the Oracle staging table for the GL Generator.

This flowchart illustrates the process:


Read Data from Oracle Retail Staging Table
Read AIA Configuration Properties
Read DVM Exports
Load Retail GL Data to EBIZ GL Interface Table
Succes
Error
Delete Source Data
Cleanup ODI Temperary Data
Commit
Rollback
Successful Notification
Error Notification

Accounting entries load process

These tasks are performed as a part of the accounting entries load process:

1. This process loads the AIAConfigurationProperties.XML file into the ODI system.

2. The system polls the STG_FIF_GL_DATA or IM_FINANCIALS_STG tables (depending on whether the RMS or ReIM package is running) until the data has been loaded in the tables.

The raten of polling depends upon a property set in the AIAConfigurationProperties.XML file.

3. ODI loads the BUSINESS_UNIT.XML and CURRENCY_CODE. XML DVM files into the system for mapping purposes.

4. The system loads all user-defined properties and variables from the AIAConfigurationProperties.XML file.

The system creates a database sequence to uniquely number entries in the PS_ORT_ACCT_ENTRY table in Oracle, if the database is not already created.

5. An ODI interface object is run to transform and map the data from the Retail table to the PSFT table.

6. If the package is successful, an email confirmation is sent to the administrator indicating the number of successfully loaded records.

The system deletes the data in the STG_FIF_GL_DATA and IM_FINANCIALS_STG tables.

Note: If any of these steps fails, an error message is sent to the AIAAsyncErrorHandlingBPELProcess and the data is rolled back into the Retail table, saving it for another run after the error is fixed.

Configuring and Running the Process Integration for Accounting Entries

This section discusses how to:

- Configure the process integration for accounting entries
- Set up a schedule
- Run the process integration for accounting entries

Configuring the Process Integration for Accounting Entries

After you have installed the process integration for accounting entries, you need to configure several parameters. Open the AIAConfigurationProperties.XML file that is available in the <AIA_HOME>/config folder.

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 *Oracle Application Integration Architecture - Foundation Pack: Core Infrastructure Components Guide,* “Using the BSR,” Loading Oracle AIA Configuration File Updates

Setting Up AIAConfigurationProperties.XML

When the process integration for accounting entries completes successfully, the system sends an email confirmation containing the number of successfully loaded records. You can configure the email properties in the AIAConfigurationProperties.XML file.

Under the module name RetailToOracle_AccountingEntry, update these properties:

```
<Property name="FromMailAddress">emailFrom</Property >
<Property name="ToMailAddress">emailTo</Property>
<Property name="MailServerName">mail.oracle.com</Property>
```

For the FromMailAddress property, configure an email address to be used as sender’s email address for email notification.

For the ToMailAddress property, configure an email address where the notification will be sent.

For the MailServerName property, configure the valid mail server which will be used to send email notifications.

For this process integration for accounting entries, the system polls the staging tables till the data has been populated in the tables. You can configure the rate of polling property in the AIAConfigurationProperties.XML file.

If desired, update the PollingInterval property:

```
<Property name="PollingInterval">60000</Property>
```

For this process integration for accounting entries, the system requires a database sequence to uniquely number entries in the PS_ORT_ACCT_ENTRY table in Oracle.

If desired, update the CommonIDSeq property:

```
<Property name="CommonIDSeq">PS_AIA_COMMON_ID_SEQ</Property>

Setting Up and Exporting DVMs

During the installation proces, the Oracle Universal Installer
promted for a location to export DVMs.  After setting up the DVMs,
export these DVMs from ESB to that location:

BUSINESS_UNIT
CURRENCY_CODE

Set up and export the CHARTOF ACCOUNTS_GLELEMENT DVM to
<AIA_HOME>/PIPS/Core/DIS/RetailToPSFTFin/CreateRetailViewSQL>.
```

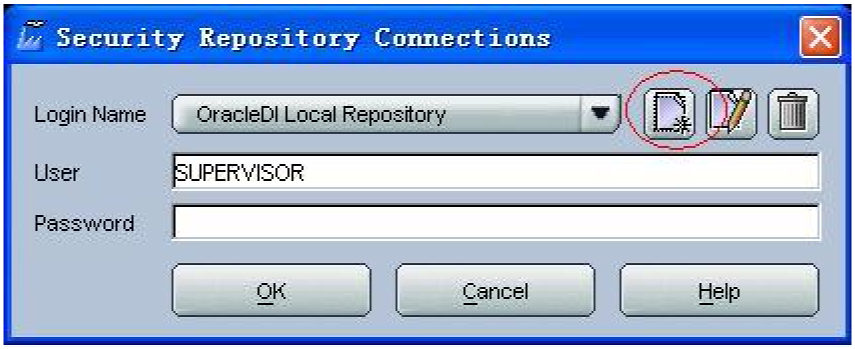
Setting Up a Schedule

Part I: Set Up Master Repository and Work Repository

Note: If you haven’t had a master repository created, you need to create it beforehand according to the ODI administrator guide.

Create connection to master repository

Launch Topology Manager, Click New.



Security Repository Connections

Login Name OracleDI Local Repository

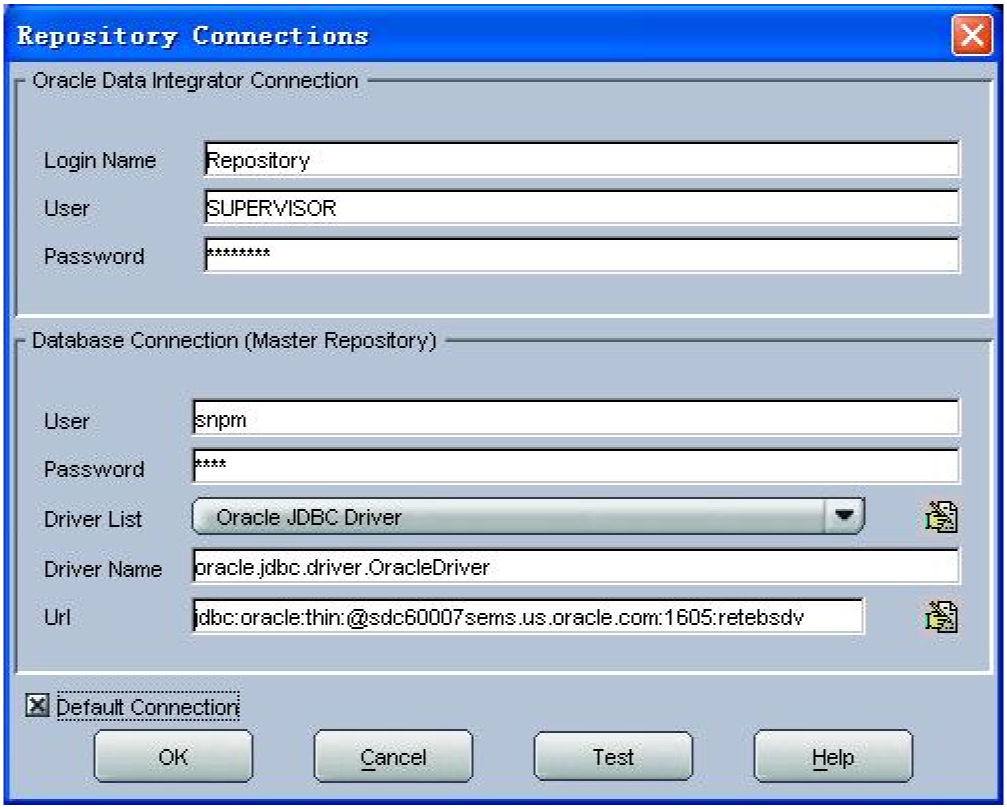
User SUPERVISOR

Password

Connect the master repository to the schema that it relies on.

Be aware that the default ODI username and password are:

SUPERVISOR/SUNOPSIS (Case sensitive)



Repository Connections

Oracle Data Integrator Connection

Login Name: Repository

User: SUPERVISOR

Password: ********

Database Connection (Master Repository)

User: snpm

Password: ****

Driver List: Oracle JDBC Driver

Driver Name: oracle.jdbc.driver.OracleDriver

Url: jdbc:oracle:thin:@sdc60007sems.us.oracle.com:1605:retebsdv

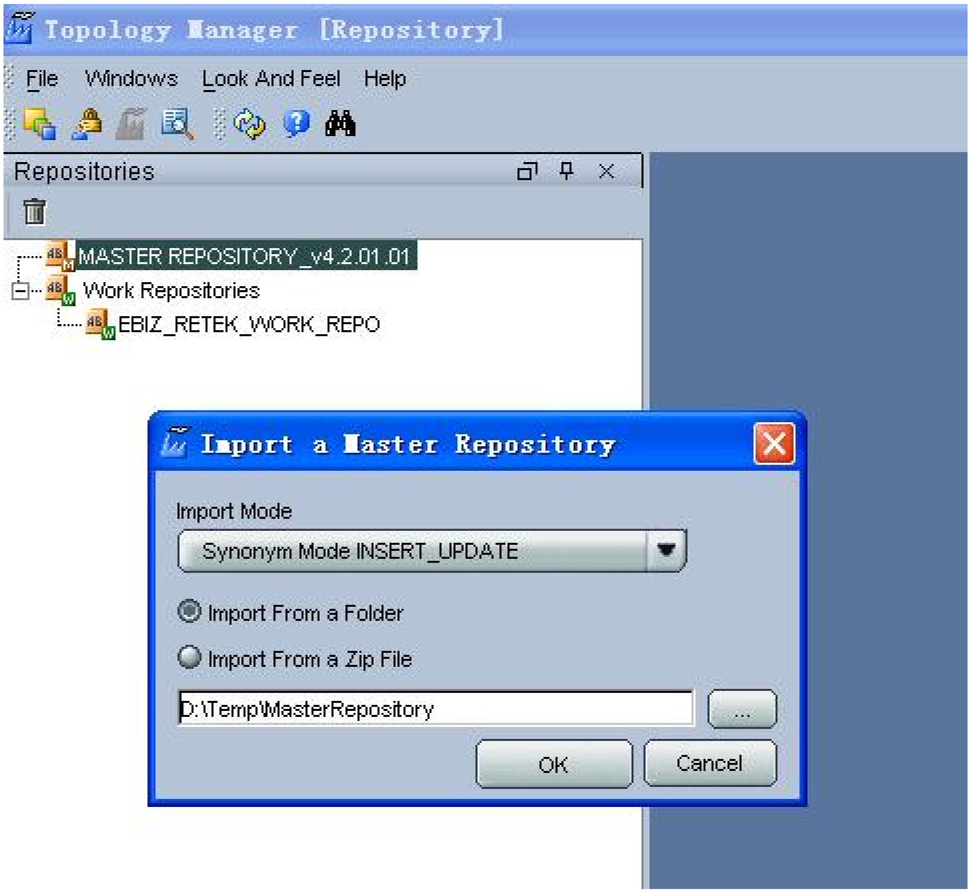
☒ Default Connection

Import master repository

The master repository files are located at

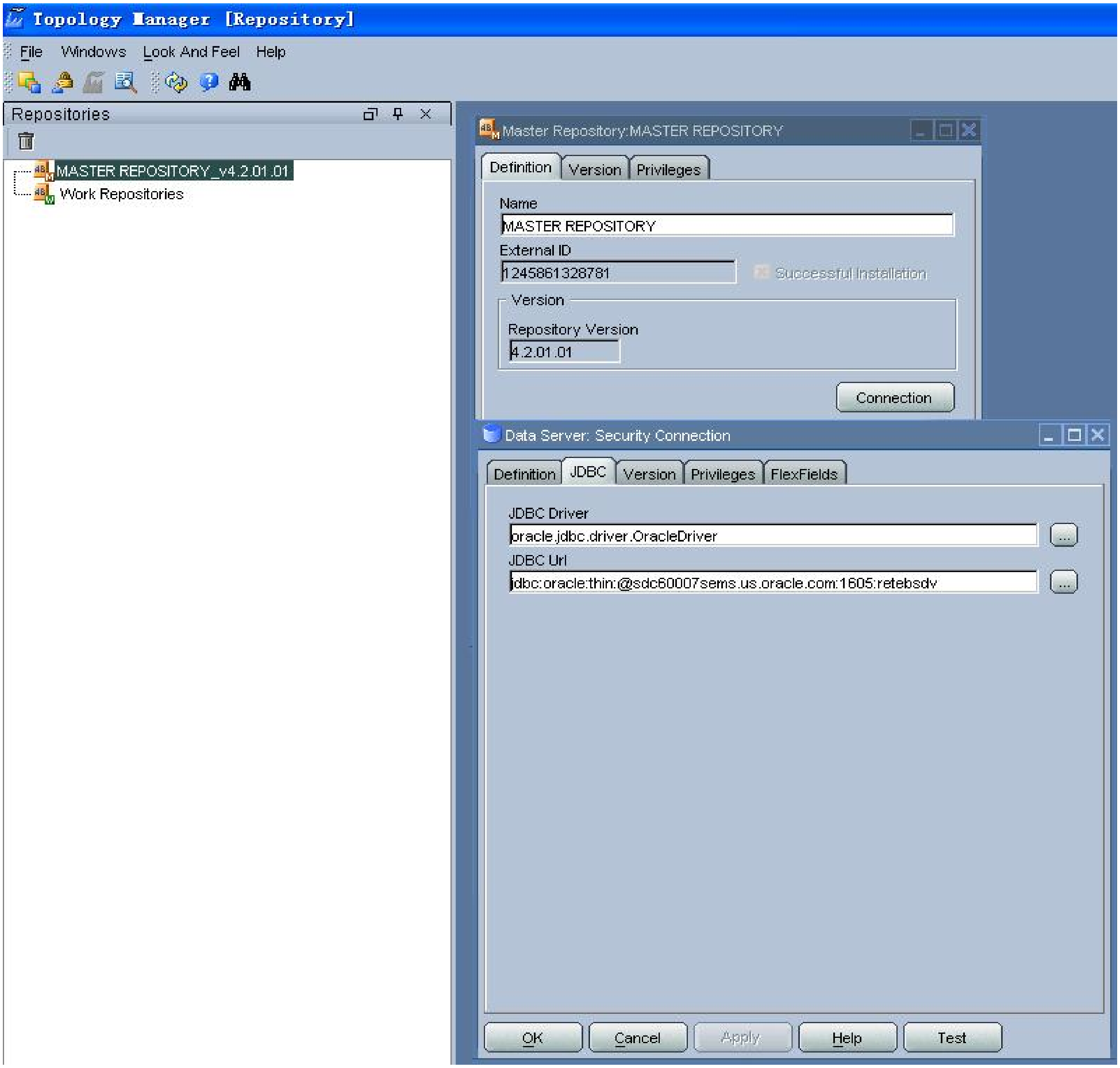
\vobs\aia\PIPS\Core\DIS\RetailToEbizFin\MasterRepository

Always choose “Synonym Mode INSERT_UPDATE”



Change master repository connection

Connect the master repository to the schema that it relies on



Name: MASTER REPOSITORY

External ID: 1245861328781

Repository Version: 4.2.01.01

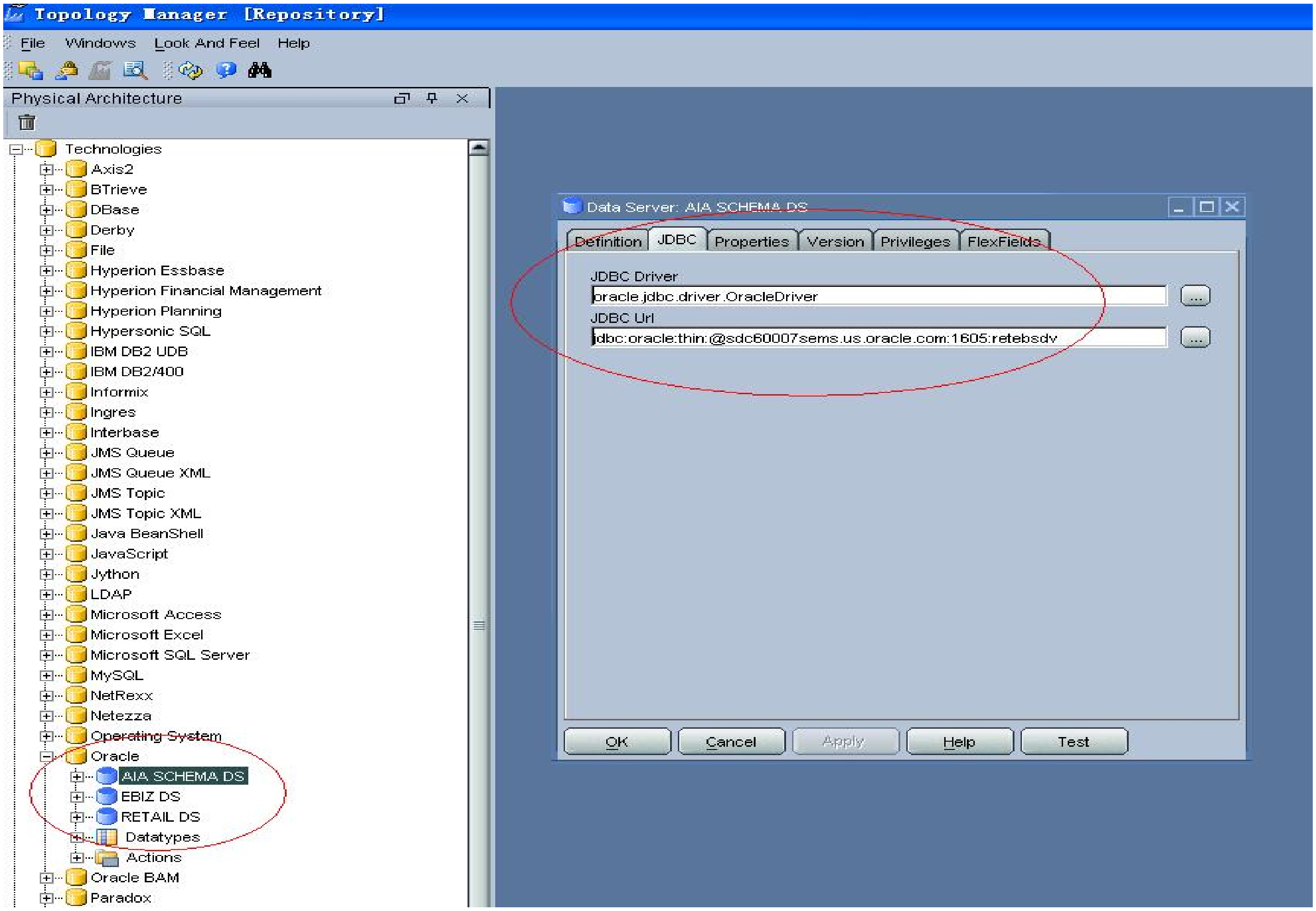
JDBC Driver: oracle.jdbc.driver.OracleDriver

JDBC Url: jdbc:oracle:thin:@sdc60007sems.us.oracle.com:1605:retebsdv

Change data source URL, username and password

Change the following data sources’ URL, username and password to connect to the instances that you are running on:

1. AIA Schema DS
2. EBIZ DS
3. RETAIL DS



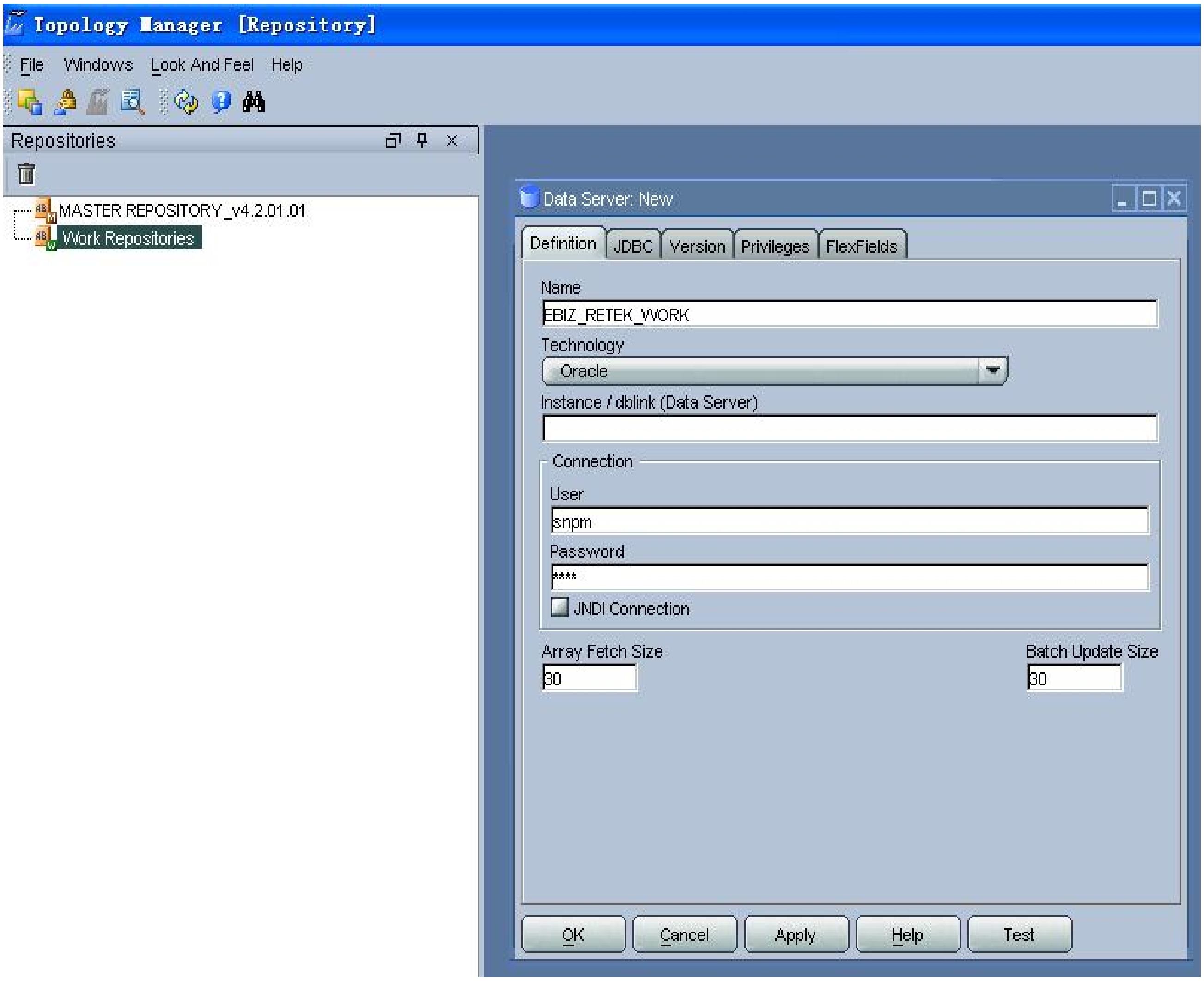
JDBC Driver

oracle.jdbc.driver.OracleDriver

JDBC Url

jdbc:oracle:thin:@sdc60007sems.us.oracle.com:1605:retebsdv

Create work repository with ID other than 900



Name: EBIZ_RETEK_WORK

Technology: Oracle

Instance / dblink (Data Server):

Connection

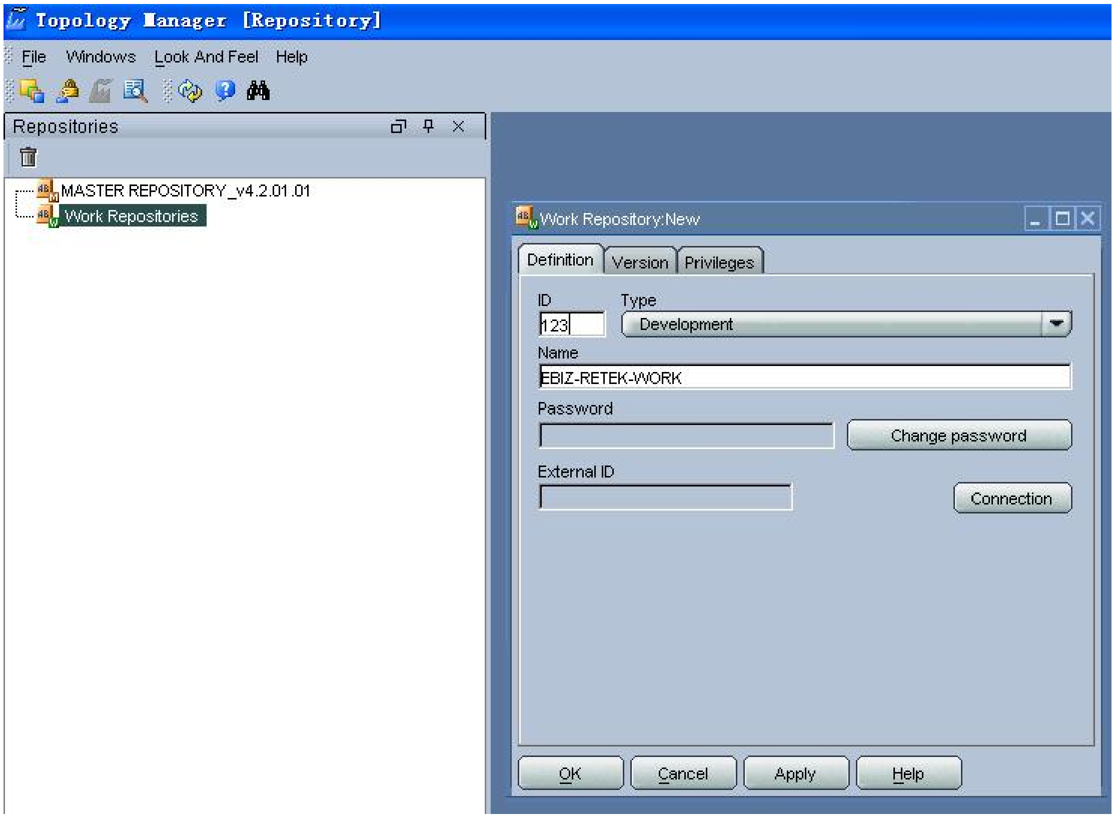
User: snpm

Password: ****

□ JNDI Connection

Array Fetch Size: 30

Batch Update Size: 30



ID: 123

Type: Development

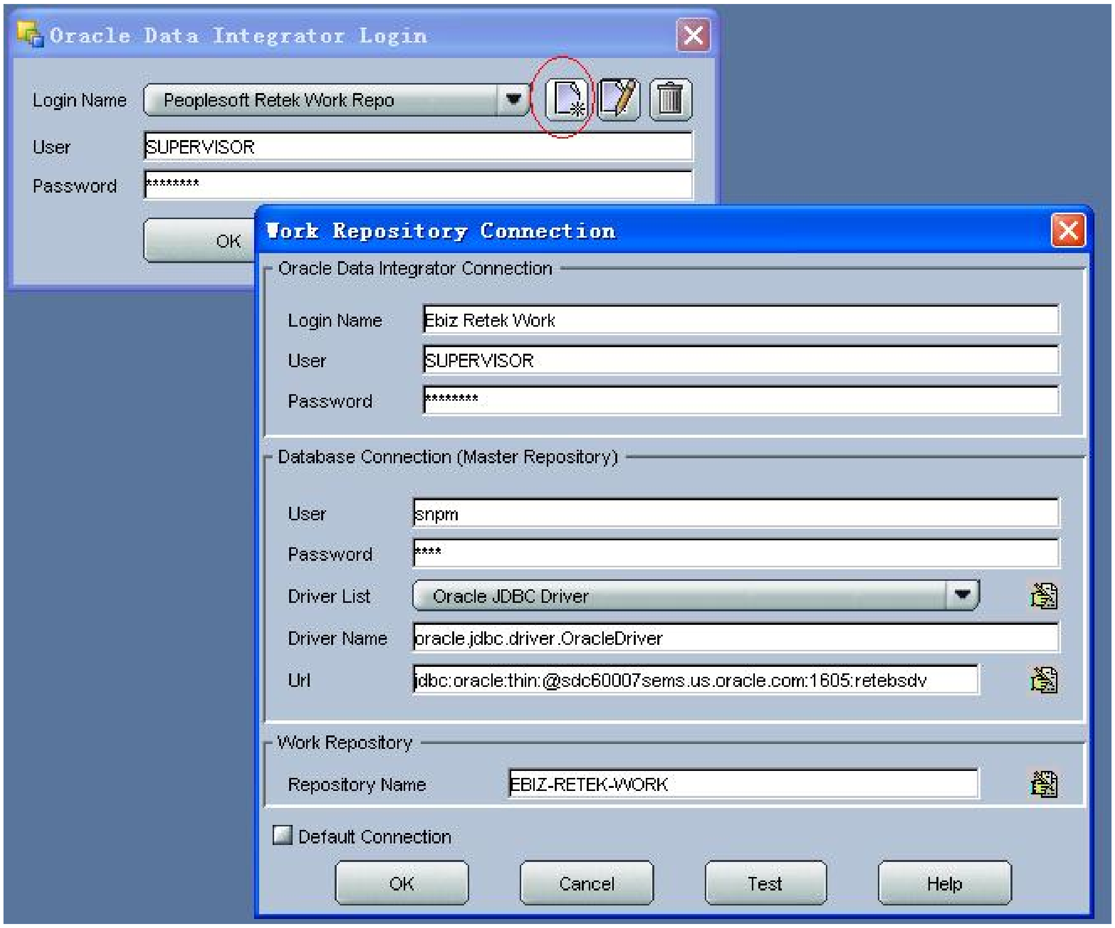
Name: EBIZ-RETEK-WORK

Password:

External ID:

Launch Designer

Launch designer, click New. Enter the work repository connection information.



Oracle Data Integrator Login

Login Name: Peoplesoft Retek Work Repo
User: SUPERVISOR
Password: ********

Work Repository Connection

Oracle Data Integrator Connection
Login Name: Ebiz Retek Work
User: SUPERVISOR
Password: ********

Database Connection (Master Repository)
User: snpm
Password: ****
Driver List: Oracle JDBC Driver
Driver Name: oracle.jdbc.driver.OracleDriver
Url: jdbc:oracle:thin:@sdc60007sems.us.oracle.com:1605:retebsdv

Work Repository
Repository Name: EBIZ-RETEK-WORK

□ Default Connection

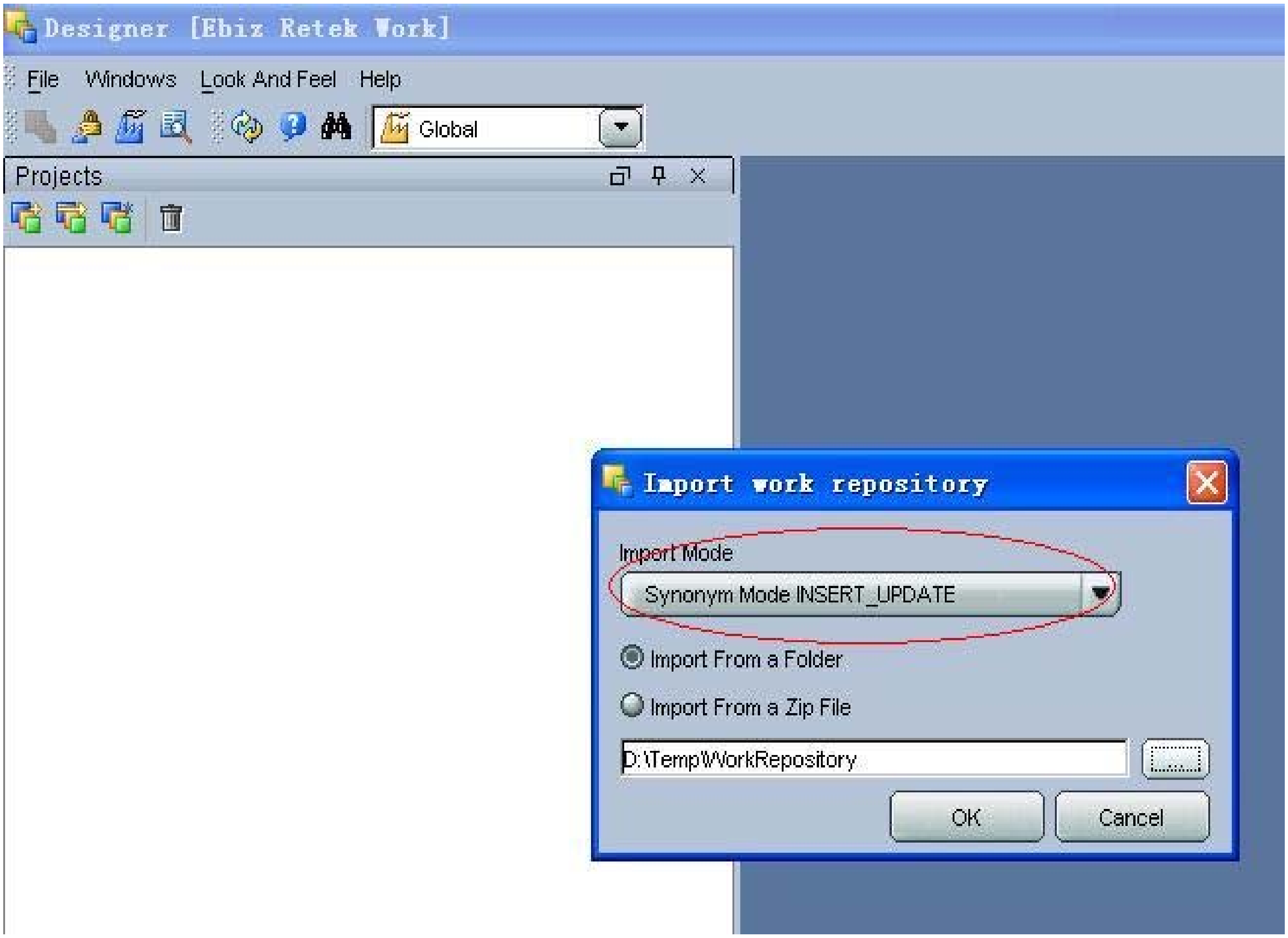
Import work repository

In the designer, click File->Import->Work Repository.

The work repository files are located at

\vobs\aia\PIPS\Core\DIS\RetailToEbizFin\WorkRepository

Always choose Synonym Mode INSERT_UPDATE.



Import Mode: Synonym Mode INSERT_UPDATE

● Import From a Folder

○ Import From a Zip File

D:\TempWorkRepository

Part II: Set up client side configuration and DVM files

In the computer where ODI is installed, create AIA Home directory, e.g. C:\AIA_HOME\config.

Copy \vobs\aia\Infrastructure\SeedData\AIAConfigurationProperties.xml from the clear case directory to the directory you created above.

Find module RetailToEbiz_AccountingEntry and

RetailToEbiz_InvoiceEntry, change the email server and addresses to the appropriate values.

Create DVM directory, for example, C:\DVM.

From the clear case directory, find and copy the following DVM files to the directory you created above:

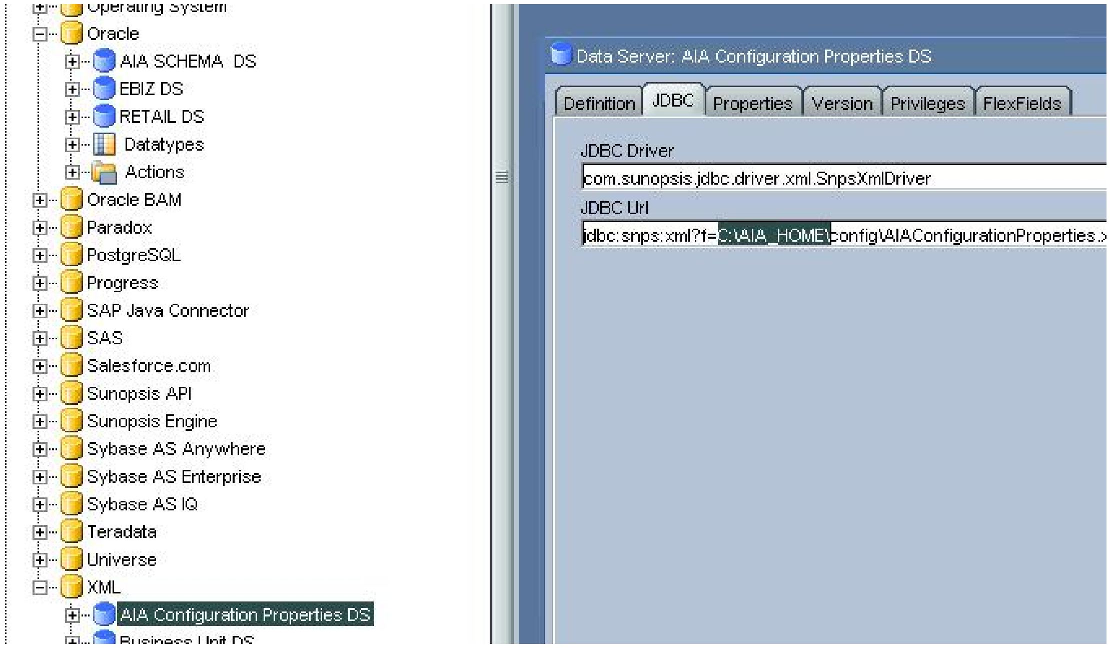
BUSINESS_UNIT.xml

CURRENCY_CODE.xml

PAYABLEINVOICE_PAYABLEINVOICELINETAXVATCODE.xml

Enter appropriate values to the DVM files.

Note: You need to check the Topology Manager -> Physical Architecture -> XML data sources to makes sure the directories are as same as the ones you created.



JDBC Driver
com.sunopsis.jdbc.driver.xml.SnpsXmlDriver

JDBC Url
jdbc:snps:xml?f=C:\AIA_HOME\config\AIAConfigurationProperties.x

Part III: Set up SOA server side XREF data

You can find the XREF files in clear case directory \vobs\aia\PIPS\Core\SeedData\XREF

Below are the XREF seed data you need to set up:

Population purpose:

- ACCOUNTINGENTRY_ACCOUNTINGENTRYID
- PAYABLEINVOICE_PAYABLEINVOICEID

Lookup purpose:

- SUPPLIERPARTY_ID
- SUPPLIERPARTY_LOCATION_ID
- ORGANIZATION_ID
- PAYMENTTERM_ID

For how to set up the XREF data, please see appendix

Part IV: Set up Ebiz source data

Login to Ebiz self-service, navigate to General Ledger responsibility, launch Categories form, enter the following categories:

Category	Key	Desc
RETEK	RETEK	RETEK
RMS	RMS	Retail RMS
Retail Invoices	Retail Invoices	Retail Invoices

Launch Sources form, enter the following sources:

Source	Key	Desc
RMS	RMS	Retail RMS
Writeoffs	Writeoffs	Retail Writeoffs
Prepayments	Prepayments	Retail Prepayments
Manual Payments	Manual Payments	Retail Manual Payments

Navigate to Payables responsibility, in the Lookups menu, launch Payables form.

Enter the following condition and search:

Type: SOURCE Application: Payables

In the result list, add the following line and save:

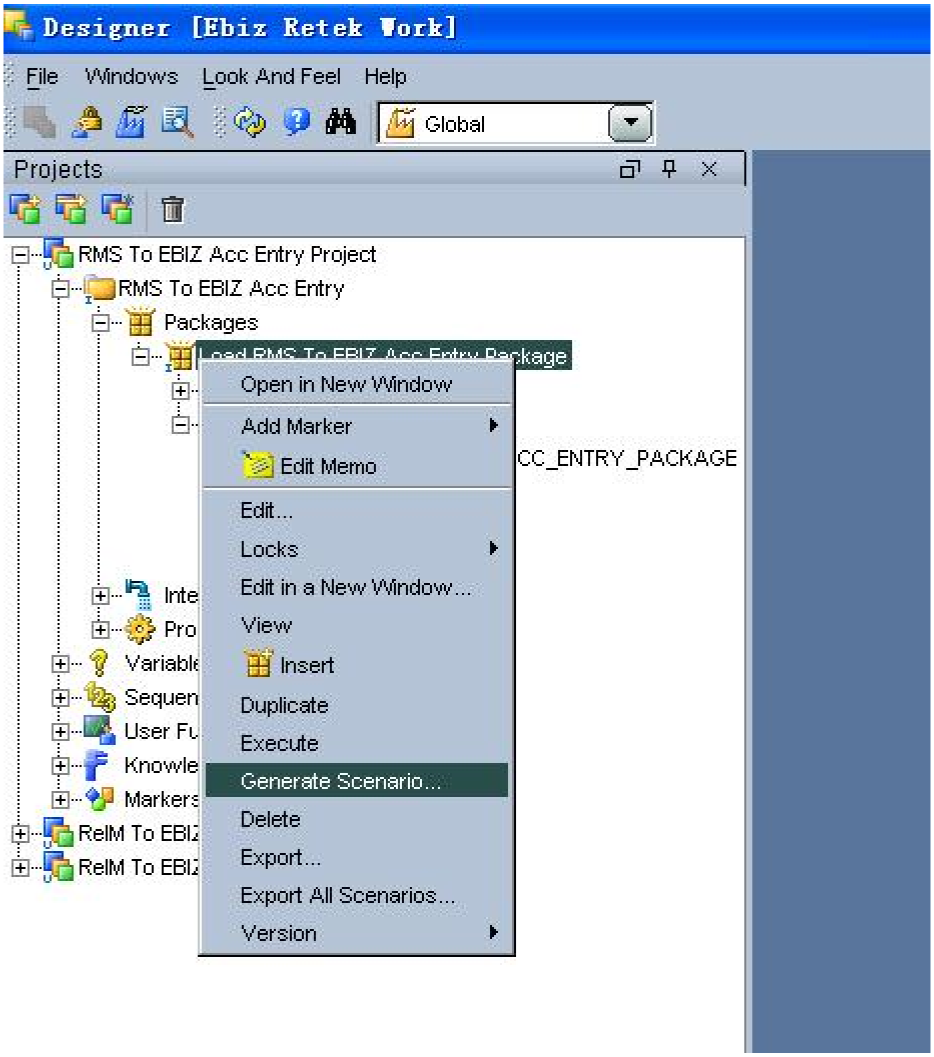
Code	Meaning	Desc
REIM	Retek ReIM	Retek ReIM

Part VI: Scenario test

In this part, you need to do similar things to the three packages in this PIP. In this guide we only use package, Load RMS To EBIZ Acc Entry Package, for example.

Generate scenario

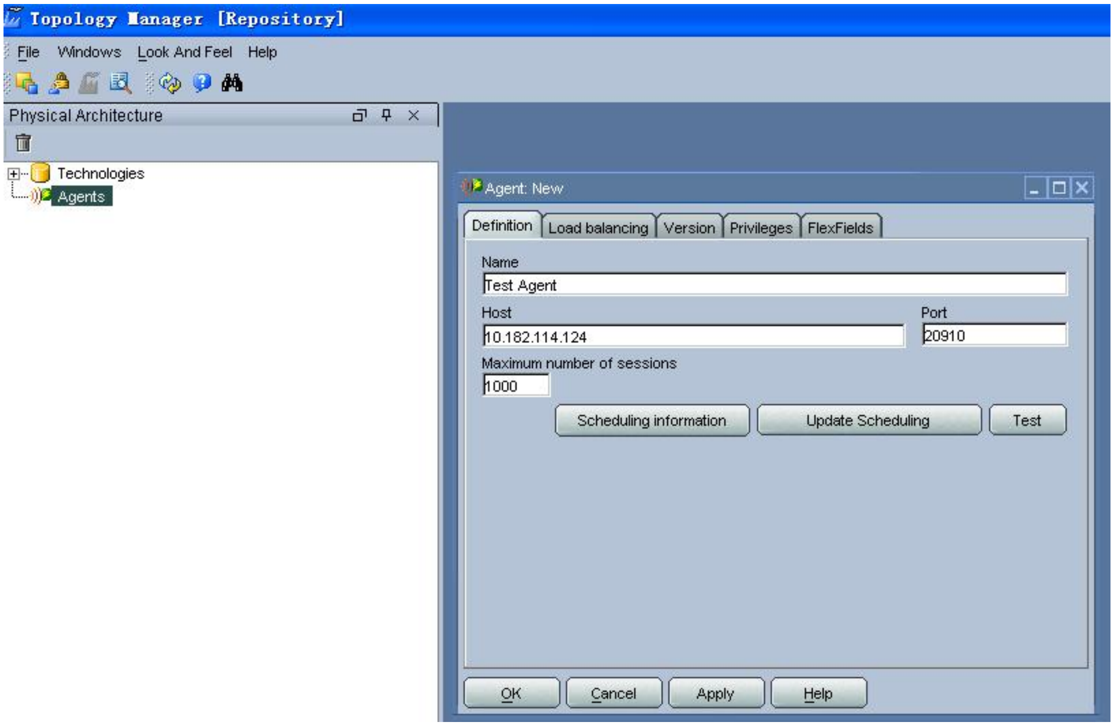
In ODI designer, navigate to the package and generate scenario, use the default values as the input.



Create physical and logical agent

This is done in the Topology Manager.

Let the port default to 20910, assign the host to the machine you are running on, name it whatever you want but remember the name.



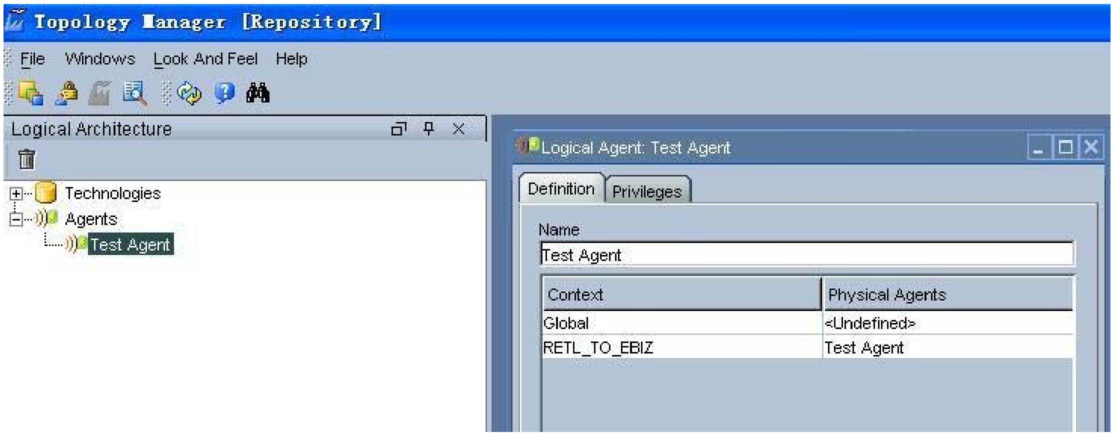
Name: Test Agent

Host: 10.182.114.124

Port: 20910

Maximum number of sessions: 1000

Create a logical agent in Logical Architecture and assign the newly created physical agent to the context RETL_TO_PSFT.



Name: Test Agent

Context	Physical Agents
Global	<Undefined>
RETL_TO_EBIZ	Test Agent

Run agent scheduler

Modify odiparams.bat file, set the following lines to connect them to the work repository, for example.

```
set ODI_SECU_DRIVER=oracle.jdbc.driver.OracleDriver

set ODI_SECU_URL=jdbc:oracle:thin:@ sdc60007sems.us.oracle.com:1605:retebsdv

set ODI_SECU_USER=snpm

set ODI_SECU_ENCODED_PASS=c6yHJ,CVjFSxK5,QqQHeALMx

set ODI_SECU_WORK_REP=EBIZ_RETEK_WORK

set ODI_USER=SUPERVISOR

set ODI_ENCODED_PASS=e3yaf23LwyAZNRyjpEWe.foc
```

You can use the following command to generate the encoded password:

```
          agent encode <password>
```

Go to a CMD prompt and type

```
      agentscheduler "-name=<physical agent name>" "-v=5"
```

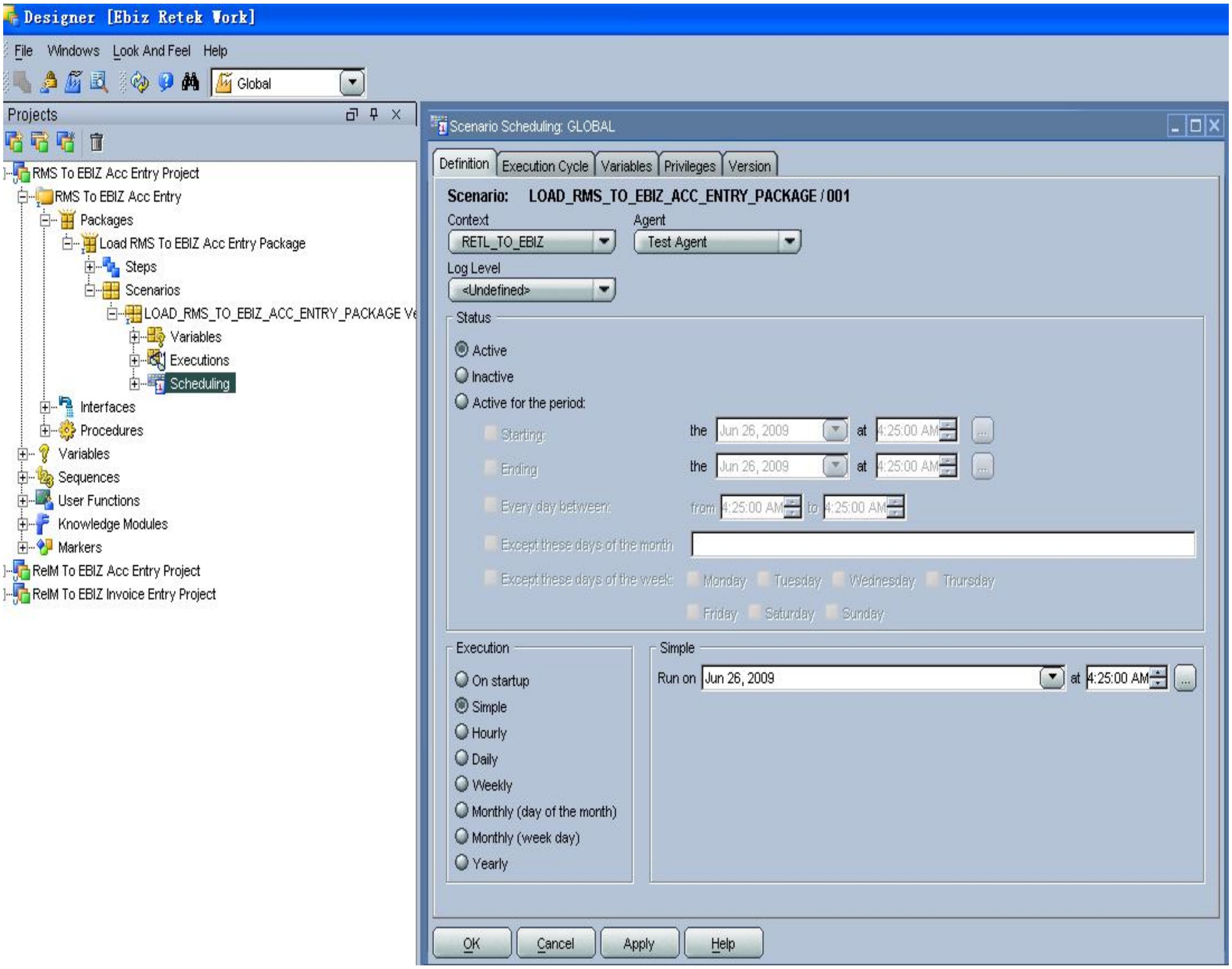
For example

```
          agentscheduler "-name=Test Agent" "-v=5"
```

You should get a message that the agent is started successfully.

Generate scheduler

Go to Designer and find the scenario you generated. Right click and insert Scheduling.



Scenario: LOAD_RMS_TO_EBIZ_ACC_ENTRY_PACKAGE / 001

Context: RETL_TO_EBIZ

Agent: Test Agent

Log Level: <Undefined>

Status

- ● Active
- ○ Inactive
- ○ Active for the period:
  - □ Starting: the Jun 26, 2009 at 4:25:00 AM
  - □ Ending: the Jun 26, 2009 at 4:25:00 AM
  - □ Every day between: from 4:25:00 AM to 4:25:00 AM
  - □ Except these days of the month
  - □ Except these days of the week: □ Monday □ Tuesday □ Wednesday □ Thursday □ Friday □ Saturday □ Sunday

Execution

- ○ On startup
- ● Simple
- ○ Hourly
- ○ Daily
- ○ Weekly
- ○ Monthly (day of the month)
- ○ Monthly (week day)
- ○ Yearly

Simple

Run on Jun 26, 2009 at 4:25:00 AM

Update schedule

Go back to Topology and find your Physical Agent and update schedule.

Check

The agent runs at a preset time. When it runs, you can check it in the Operator. You can also check the data in Ebiz side interface tables.

Run concurrent import program

When running the import program, you need to enter the source name you set up in previous steps.

When concurrent program is completed, you can check the data in SOA XREF table and Ebiz transaction tables.

New Customers to Oracle Retail to Ebiz in 2.5 Initial Load for Cross References (Post Install Directions)

Insert records into Cross-Reference table using xref utilities

To populate cross-references: (The example used below is for ORGANIZATION _ID, but can be used for any other xref table initial loads. Substitute your xref table name, column names and values.

1. Create an .xml file using the following sample. Update the XML file where the font is **BOLD** with the appropriate values:

   column name - define the columns in the table that you want to populate

   cell colName – this creates the rows and defines the values in that row for the columns

Note: the column “COMMON” is not needed in the xml file because the tool wil generate it

Sample 1 .xml: organization.xml

```
<xref xmlns="http://xmlns.oracle.com/xref">
<table name="ORGANIZATION_ID">
<columns>
 <column name="RETL_01"/>
 <column name="EBIZ_01"/>
</columns>
<rows>
 <row>
  <cell colName="RETL_01">333333333333333</cell>
  <cell colName="EBIZ_01">US001</cell>
 </row>
 <row>
  <cell colName="RETL_01">444444444444444</cell>
  <cell colName="EBIZ_01">US002</cell>
 </row>
 <row>
  <cell colName="RETL_01">555555555555555</cell>
  <cell colName="EBIZ_01">CAN01</cell>
 </row>
</rows>
</table>
</xref>
```

Sample 2 .xml: PaymentTerm_ID.xml

```
<xref xmlns="http://xmlns.oracle.com/xref">
<table name="PAYMENTTERM_ID">
<columns>
 <column name="RETL_01"/>
 <column name="EBIZ_01"/>
```

```
</columns>
<rows>
 <row>
  <cell colName="RETL_01">750</cell>
  <cell colName="EBIZ_01">30DayPay</cell>
 </row>
 <row>
  <cell colName="RETL_01">751</cell>
  <cell colName="EBIZ_01">1202</cell>
 </row>
 <row>
  <cell colName="RETL_01">752</cell>
  <cell colName="EBIZ_01">1203</cell>
 </row>
</rows>
</table>
</xref>
```

Save the xml file to a temp directory on the SOA server in the bin directory (for your particular server)

```
 $SOA_HOME/integration/esb/bin
For example:
$SOA_HOME = /slot/ems1891/oracle/product/10.1.3.1/OracleAS_1
```

NOTE: Before running the commands below, make sure that the Xref table and the columns you need to populate exist on your server. (see section **9**)

2. Make sure that you have set up your environment variables for the following, prior to running the xref tool.

   OC4J_USERNAME

   OC4J_PASSWORD

3. Run the xrefimport tool to import the cross-references by following the instructions below

4. Use Xterm or any other protocol to connect to the SOA Server.

   Xterm location:
   http://ebiztta.oraclecorp.com/cgibin/secure/ttawlogin.cgi/?action=start&pg=index2.html

5. On the command line, type the commands in the example below. Substitute your server and port

   Bold: commands

   Bold Italics: values in the command that needs to be substituted with the appropriate values

Example:

```
Log in: ssh -l ora1891 adc60117fems.us.oracle.com  (ssh –l is the
lower case letter l, not numeric 1)
```

For the login, substitute host and user name with the appropriate values for your server

You might be asked the following questions:

"Are you sure you want to continue connecting (yes/no)?" say "**yes"**

At the prompt for password type in **"welcome"** (without the quotes)

You will get the following prompt:

```
-bash-3.00$  (3.00 might be different)
```

6. At the bash prompt change directories as follows:

```
   -bash-3.00$ cd product/10.1.3.1/OracleAS_1/integration/esb/bin
    You will get the -bash prompt again
```

7. If you first need to add the xref table and columns or just the columns on the server, type: (If this is already done, move to number 10)

```
-bash-3.00$ chmod +x xreftool.sh
-bash-3.00$ ./xreftool.sh -shell
-bash-3.00$ export OC4J_USERNAME=oc4jadmin
-bash-3.00$ export OC4J_PASSWORD=welcome1
-bash-3.00$ ./xreftool.sh -shell
```

The prompt should change to >

- addTable ORGANIZATION_ID
- addColumns ORGANIZATION_ID EBIZ_01,COMMON,RETL_01

Organization_Id xref table has just been added with the columns needed for the initial load.

8. Type in the following commands to import the data in the .xml file to the xref table. This will be using the xrefimport tool. After each commnad, you will be returned to the –bash prompt

```
-bash-3.00$ chmod +x xrefimport.sh
-bash-3.00$ ./xrefimport.sh -shell
```

9. Set the following environment variables with the correct values, using the script below:

```
            DB_USER
            DB_PASSWORD
            OC4J_USERNAME
            OC4J_PASSWORD
            DB_URL

      Example:
      -bash-3.00$ export DB_USER=aia
      -bash-3.00$ export DB_PASSWORD=aia
      -bash-3.00$ export OC4J_USERNAME=oc4jadmin
      -bash-3.00$ export OC4J_PASSWORD=welcome1

      (the next three lines need to go in one line)
      -bash-3.00$ export
            DB_URL="jdbc:oracle:thin:@adc60117fems.us.oracle.
                  com:1612:finandb"
```

(substitute the following with values for your server)

```
db hostname - adc60117fems.us.oracle
jdbc port - 1612
sid - finandb
```

10. Import the data using the command below:

```
   -bash-3.00$ xrefimport.sh -file \TEMP/organization_id.xml  -mode
         ignore -generate COMMON
Parameters:
   -file = your xml file name with the directory you saved it in
(your file should be in a folder in the bin directory) Since you are
already in the bin directory, you don’t need to change dir Required)
   -mode = defines how to handle conflicts with existing data in the
xref table. There are two possibilities for mode (Optional)
   -mode ignore = existing records in the xref table will be ignored
or left as is.  New data from the xml file will be appended to the
existing data if different.  If the data is a duplicate for a row, a
new record will not be written.
   -mode overwrite = if there is a conflict with the data, the
   existing data in question will be overwritten with the new data
from the XML file
    -generate COMMON = this tells the tool to generate a common
record for each set of rows with an auto-generated guid. It links
the records together by row number (Optional)
```

Validating Cross-References

To validate cross-references:

1. Log on to the AIA database

2. Query the Table XREF_DATA to confirm that every organization used in the XML files has three records.

   Use the following query:

```
The query below will retrieve all records in the ORGANIZATION_ID
TABLE
select * FROM xref_data where xref_table_name='ORGANIZATION_ID'


or


  The query below will retrieve the three associated records for the
   org_id you selected
   select * from xref_Data
   where row_number in (select row_number from xref_Data
   where value = 'CAN01')
```

Replace the value for the organizations you selected. (The number of operating units depends on your setup.)

3. Validate that for each row, three records were written.

RETL_01, EBIZ_01 and COMMON

The three records should be linked to each other with a common row number value

For more information about creating schedules and scenarios, see *Oracle Data Integrator User Guide* and *Oracle Data Integrator Installation Guide*. These documents are available in <odi install location?>\oracledi\doc subdirectory.

Running the Process Integration for Accounting Entries

This process runs based on the schedules created for it. The integration between Oracle Retail and Oracle GL can be scheduled to occur automatically.

Note: You can start or stop the scheduler agent whenever you want, which enables you to control the integration process.

For more information about starting and stopping the scheduler agent, see *Oracle Data Integrator User Guide*, “Launching a Scheduler Agent” and “Stopping an Agent.” This document is available in <odi install location?>\oracledi\doc subdirectory

Chapter 5: Process Integration for Retail Merchandise Procure to Pay

This chapter provides an overview of the process integration for Retail Merchandise Procure to Pay between Oracle Retail Invoice Matching (ReIM) and Oracle Payables and discusses how to:

- Perform setup tasks
- Configure and generate data in Oracle Retail
- Pick up and transform the data
- Configure and run the process integration for Retail Merchandise Procure to Pay

Process Integration for Retail Merchandise Procure to Pay Overview

ReIM is the source of matched invoices, credit notes, debit memos, and rebates. Oracle Payables requires these details for payment to the suppliers. Invoices from suppliers for retail merchandise are matched to the original purchase order (PO) for the merchandise and the receipt of the merchandise by the retailer. A proper match of invoice, PO, and receipt trigger the payment authorization of the supplier's invoice. Invoices may be authorized for payment prior to receipt of goods for which prepayment is required. When the authorization for payment is generated, the appropriate accounting distribution is also generated to support the payment authorization. The Retail Merchandise Procure to Pay integration automates the processing of invoice payments, adjustments, and write-offs from ReIM to Oracle Payables and General Ledger. Other accounting transactions are generated from ReIM to write off aged receipts that were never invoiced and to post accounting distribution for manually paid or prepaid invoices after receipt.

This integration uses the Oracle Data Integrator (ODI) application to transfer the data from ReIM to Oracle Payables.

The process integration between ReIM and Oracle Payables supports these integration flows:

- Post matched invoices for payment from ReIM to Oracle Payables.
- Post credit notes (matched or unmatched) for payment adjustment from ReIM to Oracle Payables.
- Post debit or credit memos for payment adjustment from ReIM to Oracle Payables.
- Post rebates for payment adjustment from ReIM to Oracle Payables.
- Post unmatched invoices for prepayment from ReIM to Oracle Payables.

Business Process Flow for Integration of Retail Merchandise and AP_INVOICE_LINES_INTERFACE. Correct the PeopleSoft Fin columns to reflect ebs tables and correct column heading.

Work Location	Step

Work Location	Step
Oracle Retail (ReIM)	1. Configure and generate outgoing data from ReIM.
Integration process	2. The integration process checks for data in the ReIM Invoice Header and Invoice Lines interface tables. 3. Populate the Invoice cross-reference with the Retail key and common key. 4. Load and transform the Oracle Retail data into the Oracle Voucher Header and Voucher Lines Staging tables. 5. Delete the data from the source tables (ReIM Invoice Header and Invoice Lines interface tables) after a successful load.
Oracle Payables	6. Run the Voucher Build process and import the vouchers into Oracle Payables system for the pay cycle. 7. Invoke a process to populate the new Oracle Invoice XREF table with the PSFT key and RETL key.

Prerequisites

Before performing this process integration, ensure that:

1. Chart of accounts code combinations are validated.
2. Calendar must be set up for transactions.
    a. Supplier synchronization and supplier cross-reference.
    b. Supplier address synchronization and supplier address cross-reference.
    c. Payment terms synchronization and payment terms cross-reference.
    d. Currency code domain value mapping (DVM) and currency exchange rate synchronization.
3. Chart of accounts combinations are manually entered into the Oracle Retail and E-Business Suite applications.
4. The DVM data is set up in the Service Oriented Architecture (SOA) suite and exported from the SOA suite that is used as input in the ODI interfaces.

Assumptions and Constraints

The integration design assumes that:

1. The data moves from one instance of source database to one instance of target database.
2. If the user loads the same data twice in the Oracle Retail interface table then the data is transported again to the Oracle interface tables.

   In the ODI interface, no business validation is applied to check whether any data is already transported.

3. The chart of accounts combinations are valid for invoice lines.

Invoice lines with invalid combinations are rejected by General Ledger and the correction is a manual process in E-Business Suite applications.

4. Oracle GL does not delete the invoices with source as RETL.
5. Oracle Retail modifies the IM_AP_STAGE_HEAD and IM_AP_STAGE_DETAIL tables and adds an ORG_UNIT column to map it to BUSINESS_UNIT (AP).

Performing Setup Tasks

This section discusses:

- Setup tasks specific to ReIM
- Setup tasks specific to Oracle General Ledger
- Setup tasks specific to Oracle Payables
- Setup tasks specific to the integration layer

Setup Tasks Specific to ReIM

For more information, see the *ReIM Operations Guide* for details about the setup tasks specific to ReIM.

Setup Tasks Specific to Oracle Payables

Perform these tasks

- Apply patch 7567527:R12.AP.B and 8502340:R12.ZX.B
- Enable event oracle.apps.ap.invoice.import if customer wants the AIA XREF to be populated
- Suppliers synchronization
- Supplier locations synchronization
- Payment Terms synchronization
- Chart of account code combination validation
- Currency Exchange Synchronization
- The invoice source has been setup in E-Business Suite.
- The following DVM have been entered on the ESB console:
    - CURRENCY CODE
    - PAYABLE INVOICE TAX CODE
- The following AIA XREF has been populated
    - ORGANIZATION_ID

- SUPPLIERPARTY_ID
- SUPPLIERPARTY_LOCATION_ID
- PAYMENTTERM_ID

Setup Tasks Specific to the Integration Layer

Setup tasks specific to this integration layer

- Perform these tasks:
  - EXPORT DVM

    During install process, the Oracle Universal Installer will prompt for a location to export the DVMs. Use that location to export the following DVMs from ESB:
  - CURRENCY CODE
  - PAYABLE INVOICE TAX CODE
  - Set up the customer email address and data pooling time in AIA Configuration properties XML file. This file exist on <AIA_HOME>/Infrastructure/SeedData. We'll add a ModuleConfiguration element to this file, RetailToE-BusinessSuiteInvoiceEntry for invoice entry. See appendix sample data for more.

Configuring and Generating Data in Oracle Retail

This section discusses how to:

- Configure the data in Oracle Retail
- Generate the data in Oracle Retail

Configuring the Data in Oracle Retail

Before running any transaction data, these tasks must be done in ReIM:

- General ledger (GL) account cross-reference is set up.
- GL options are defined.

For more information, see the *ReIM User Guide,* "System Administration," General Ledger Accounts

Generating the Data in Oracle Retail

In ReIM, the batch process engages in these high-level steps:

1. Performs any resolution actions (for example, instigates the creation of payment documents).

2. Calls the posting process to write applicable financial accounting transactions to the financials staging table, IM_FINANCIALS_STAGE.

The processing occurs after discrepancies for documents have been resolved by resolution documents. Once all of the resolution documents for a matched invoice are built, and all of the RCA/RUA external processing has been confirmed, the process inserts financial accounting transactions to the financials staging table, to represent the resolution and consequent posting of the invoice. The process also inserts financial accounting transactions for the approved documents that are being handled.

Once all of the transactions have been written, the process switches the status of the current invoices or documents to "Posted", and then moves on to the next invoice or document. If a segment look-up fails, the failed record is written to a financials error table.

For more information, see the *ReIM Operations Guide,* "Batch Processes," Resolution posting action rollup batch design

Configuring and Running the Process Integration for Retail Merchandise Procure to Pay

This section discusses how to:

- Set up a schedule

Configuring the Process Integration for Retail Merchandise Procure to Pay

After you have installed the process integration for Retail Merchandise Procure to Pay, you need to configure several parameters. Open the AIAConfigurationProperties.XML file that is available in the <AIA_HOME>/config folder.

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 *Oracle Application Integration Architecture - Foundation Pack: Core Infrastructure Components Guide,* "Using the BSR," Loading Oracle AIA Configuration File Updates

Prerequisite

Verify that Process Integration Pack (PIP) ODI Artifacts (code) has been imported and connections are configured.

Setting Up AIAConfigurationProperties.XML

When the process integration for Retail Merchandise Procure to Pay completes successfully, the system sends an email confirmation containing the number of successfully loaded records. You can configure the email properties in the AIAConfigurationProperties.XML file.

Under the moduleName RetailToOracleInvoiceIntegration, update these properties:

```
<Property name="FromMailAddress">emailFrom</Property>
<Property name="ToMailAddress">emailTo</Property>
<Property name="MailServerName">mail.oracle.com</Property>
```

For the FromMailAddress property, configure an email address to be used as sender's email address for email notification.

For the ToMailAddress property, configure an email address where the notification will be sent.

For the MailServerName property, configure the valid mail server which will be used to send email notifications.

For this process integration, the system polls the staging tables till the data has been populated in the tables. You can configure the rate of polling property in the AIAConfigurationProperties.XML file.

If desired, update the PollingInterval properties:

```
<Property name="PollingInterval">60000</Property>
```

For this process integration, the system requires a database sequence to uniquely number entries in the PS_ORT_ACCT_ENTRY table in Oracle.

If desired, update the CommonIDSeq property:

```
<Property name="CommonIDSeq">PS_AIA_COMMON_ID_SEQ</Property>
```

Additional step required for supplier

1. change the value of BypassAddressIDXref at file "AIAConfigurationProperties.xml"
   <Property name="BypassAddressIDXref">**YES**</Property>

Setting Up and Exporting DVMs

Set up these DVMs on the ESB console:

- BUSINESS_UNIT
- CURRENCY_CODE
- CURRENCYEXCHANGE_CONVERSIONTYPECODE

During the installation process, the Oracle Universal Installer prompts for a location to export the DVMs. Export these DVMs from ESB to that location.

> **For more information** about how to create DVMs, see *Oracle Enterprise Service Bus Developer's Guide*, "Creating and Populating Domain-Value Maps"

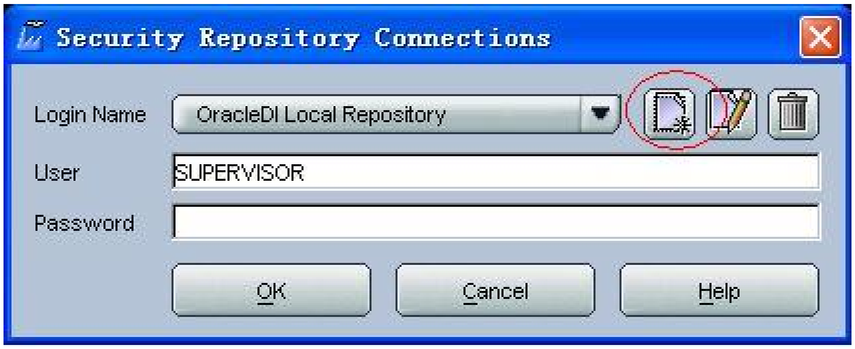
Setting Up a Schedule

Part I: Set up master repository and work repository

Note: If you haven't had a master repository created, you need to create it beforehand according to the ODI administrator guide.

Create connection to master repository

Launch Topology Manager. Click “New”



Security Repository Connections

Login Name: OracleDI Local Repository

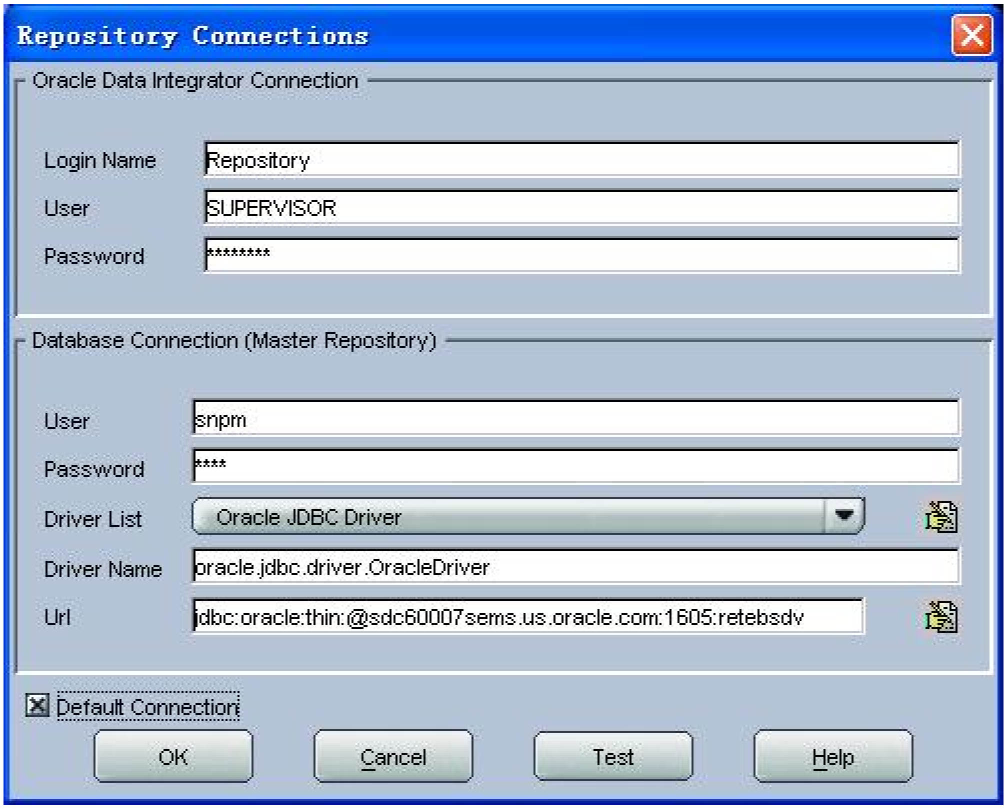
User: SUPERVISOR

Password:

Connect the master repository to the schema that it relies on.

Be aware that the default ODI username and password are:

SUPERVISOR/SUNOPSIS (Case sensitive)



Repository Connections

Oracle Data Integrator Connection

Login Name: Repository

User: SUPERVISOR

Password: ********

Database Connection (Master Repository)

User: snpm

Password: ****

Driver List: Oracle JDBC Driver

Driver Name: oracle.jdbc.driver.OracleDriver

Url: jdbc:oracle:thin:@sdc60007sems.us.oracle.com:1605:retebsdv

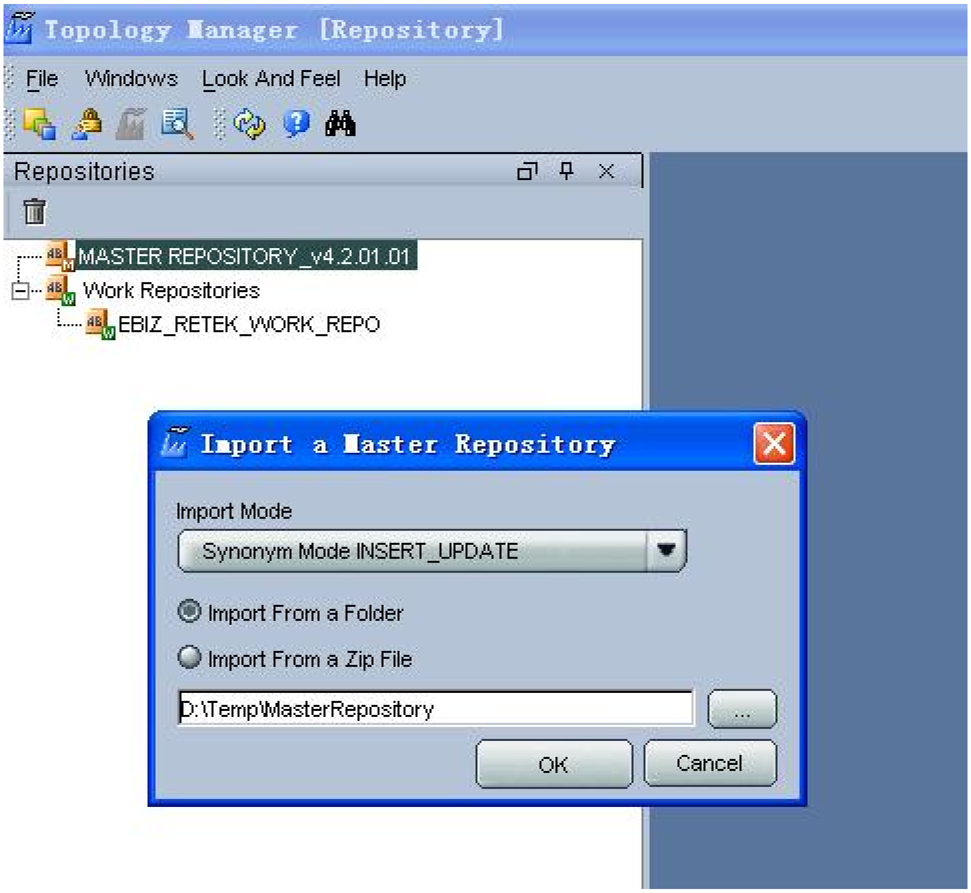
☒ Default Connection

Import master repository

The master repository files are located at

\vobs\aia\PIPS\Core\DIS\RetailToEbizFin\MasterRepository

Always choose “Synonym Mode INSERT_UPDATE”



Import Mode

Synonym Mode INSERT_UPDATE

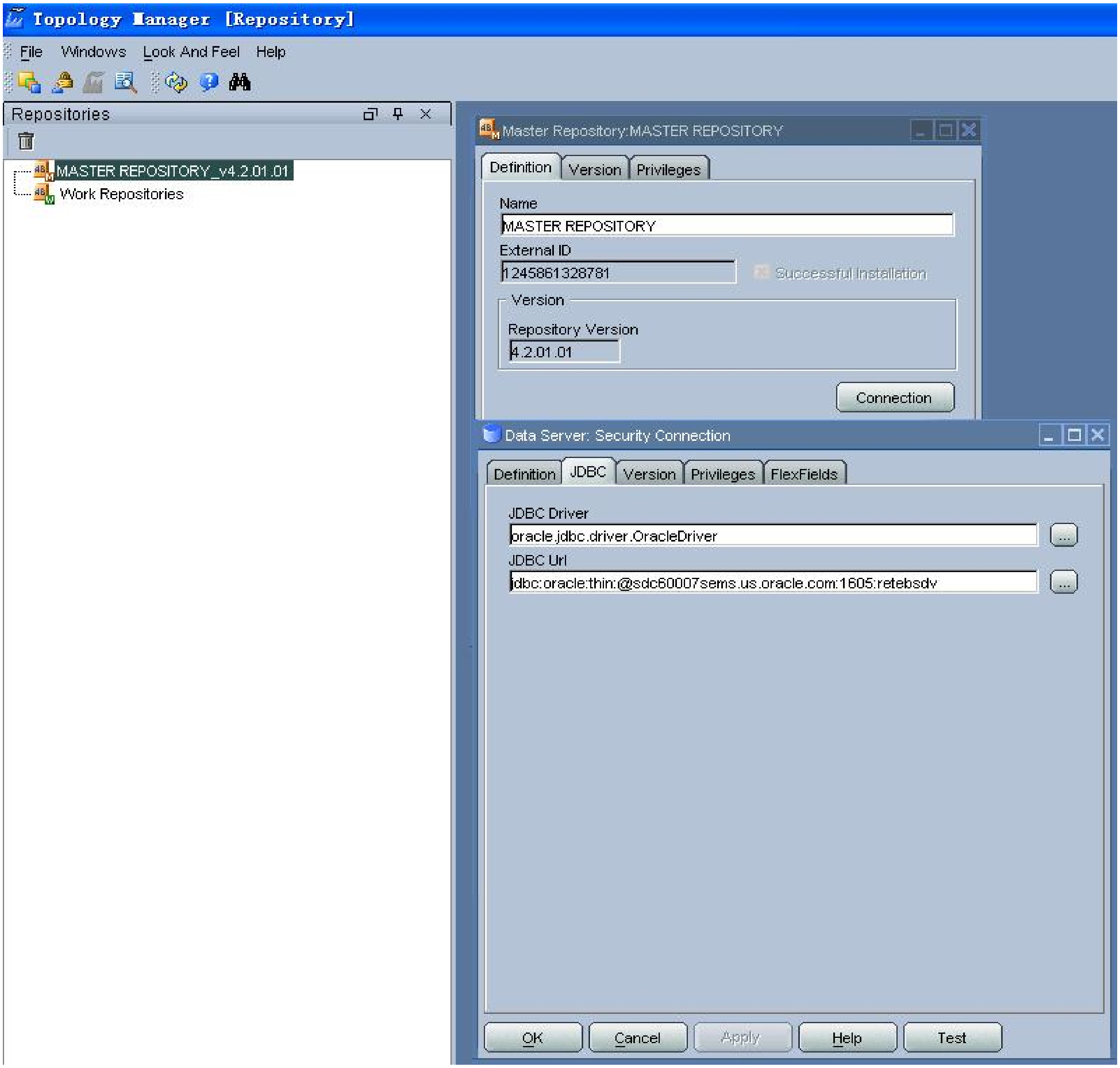
● Import From a Folder

○ Import From a Zip File

D:\Temp\MasterRepository

Change master repository connection

Connect the master repository to the schema that it relies on



Name
MASTER REPOSITORY

External ID
1245861328781

Repository Version
4.2.01.01

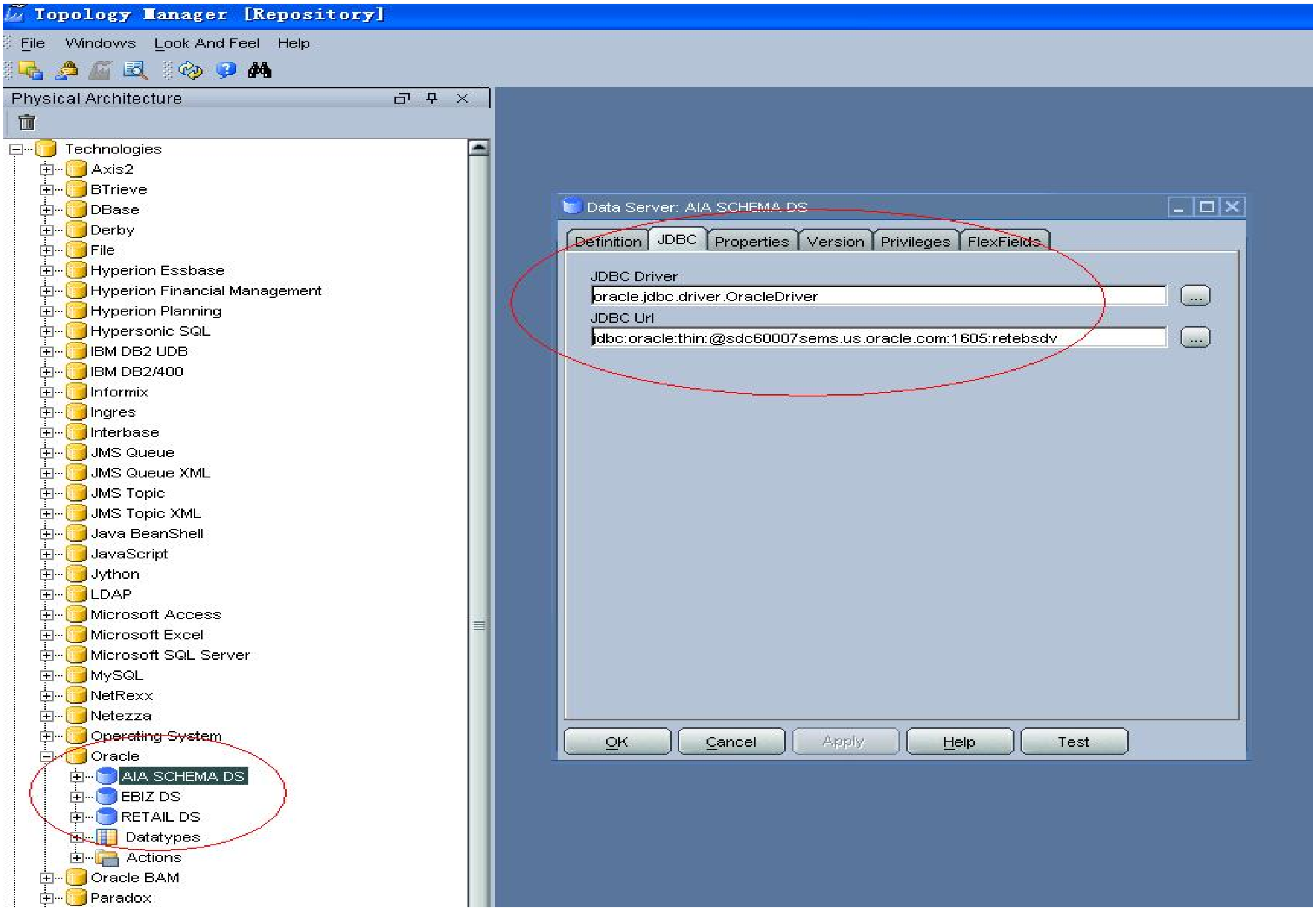
JDBC Driver
oracle.jdbc.driver.OracleDriver

JDBC Url
jdbc:oracle:thin:@sdc60007sems.us.oracle.com:1605:retebsdv

Change data source URL, username and password

Change the following data sources' URL, username and password to connect to the instances that you are running on:

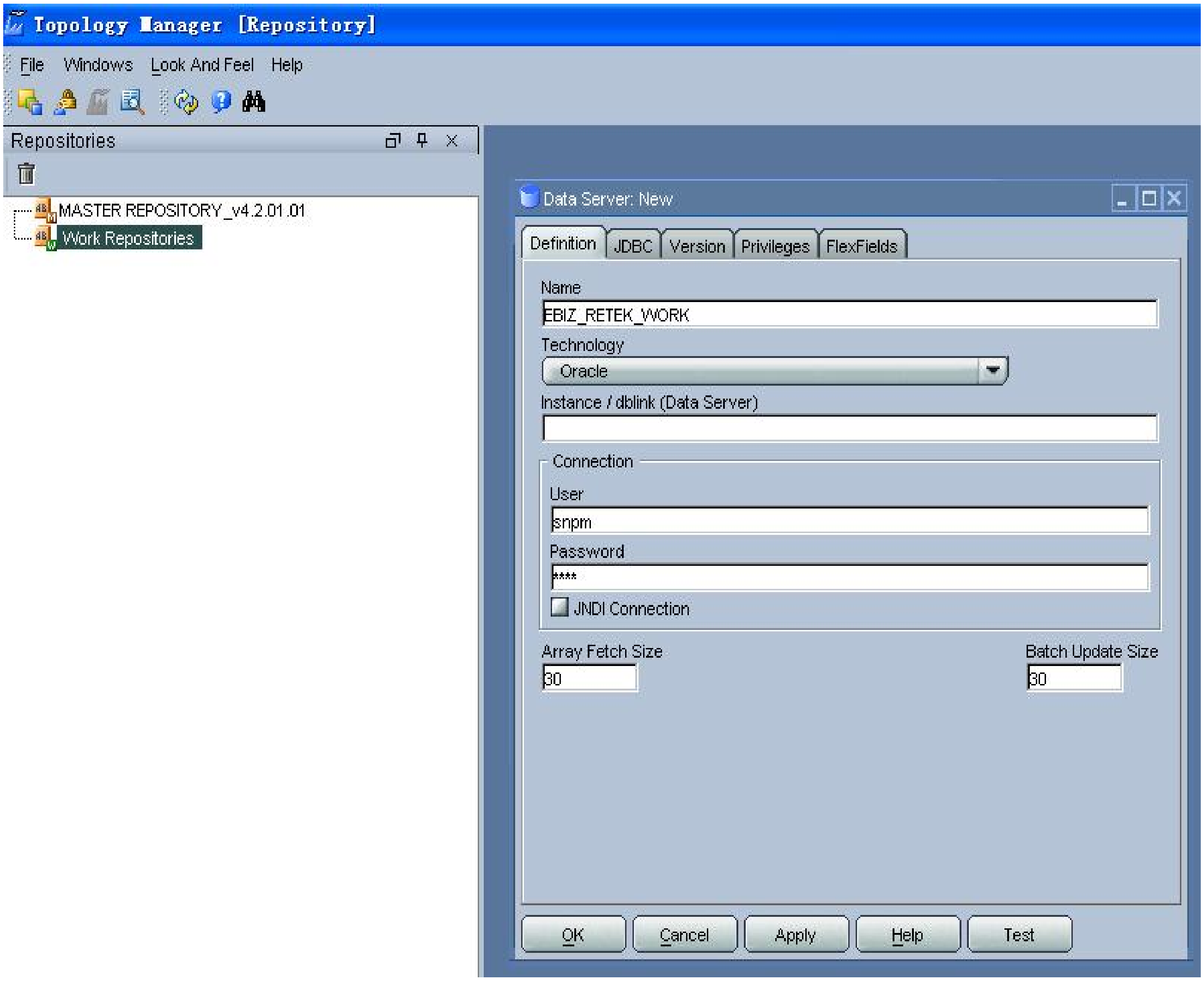
1. AIA Schema DS
2. EBIZ DS
3. RETAIL DS



JDBC Driver: oracle.jdbc.driver.OracleDriver

JDBC Url: jdbc:oracle:thin:@sdc60007sems.us.oracle.com:1605:retebsdv

Create work repository with ID other than 900



Name: EBIZ_RETEK_WORK

Technology: Oracle

Instance / dblink (Data Server):

Connection

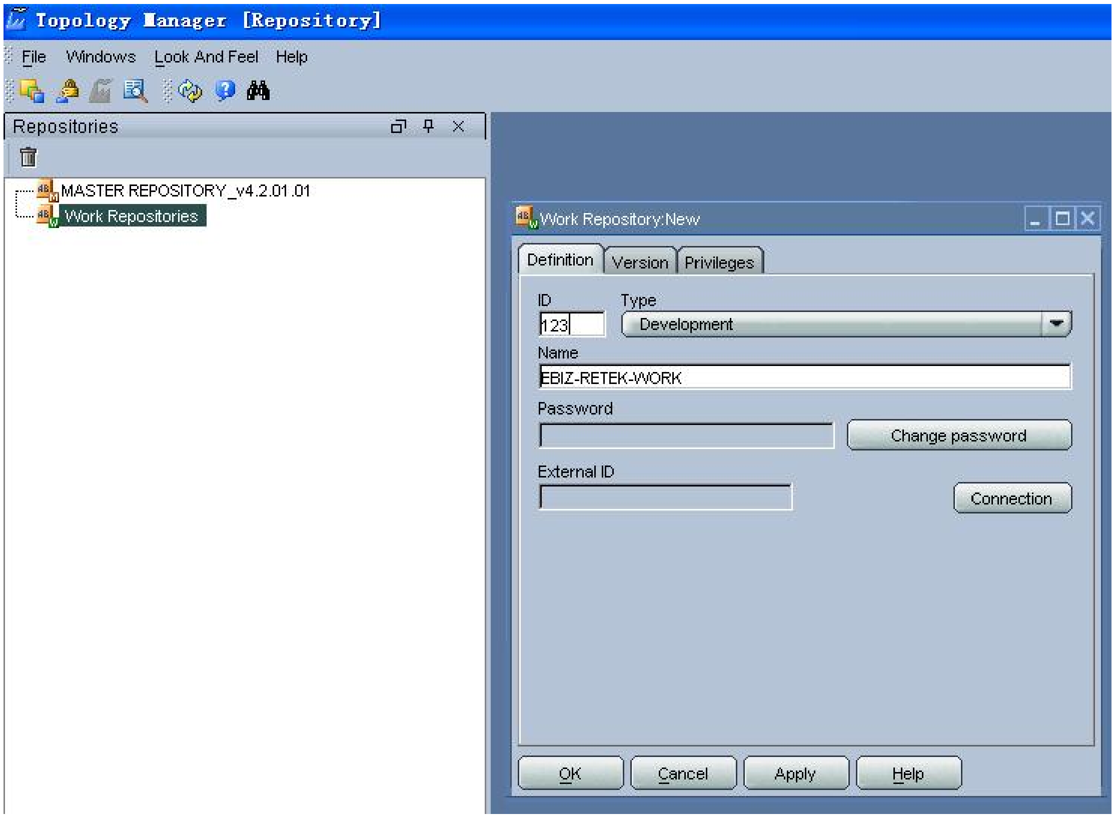
User: snpm

Password: ****

□ JNDI Connection

Array Fetch Size: 30

Batch Update Size: 30



ID: 123

Type: Development

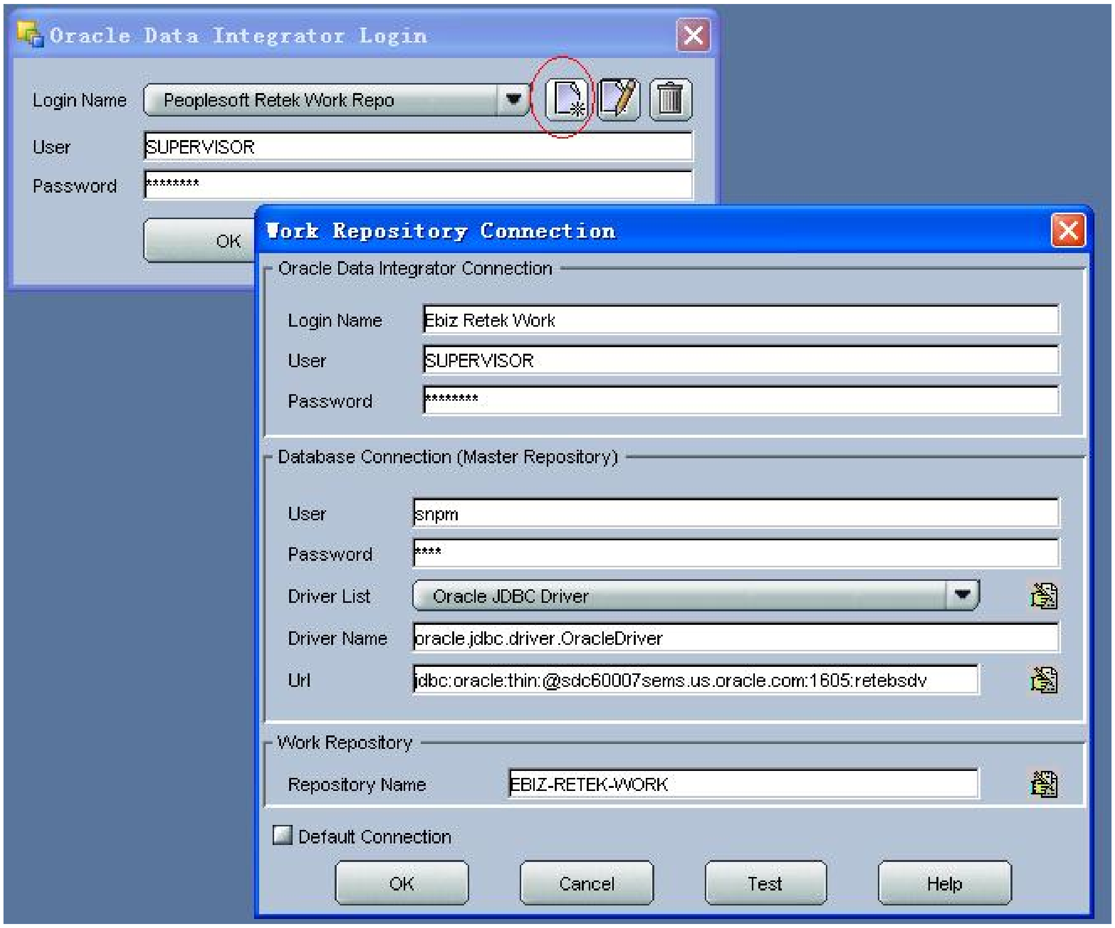
Name: EBIZ-RETEK-WORK

Password:

External ID:

Launch Designer

Launch designer, click New, enter the work repository connection information.



Oracle Data Integrator Login

Login Name: Peoplesoft Retek Work Repo

User: SUPERVISOR

Password: ********

Work Repository Connection

Oracle Data Integrator Connection

Login Name: Ebiz Retek Work

User: SUPERVISOR

Password: ********

Database Connection (Master Repository)

User: snpm

Password: ****

Driver List: Oracle JDBC Driver

Driver Name: oracle.jdbc.driver.OracleDriver

Url: jdbc:oracle:thin:@sdc60007sems.us.oracle.com:1605:retebsdv

Work Repository

Repository Name: EBIZ-RETEK-WORK

□ Default Connection

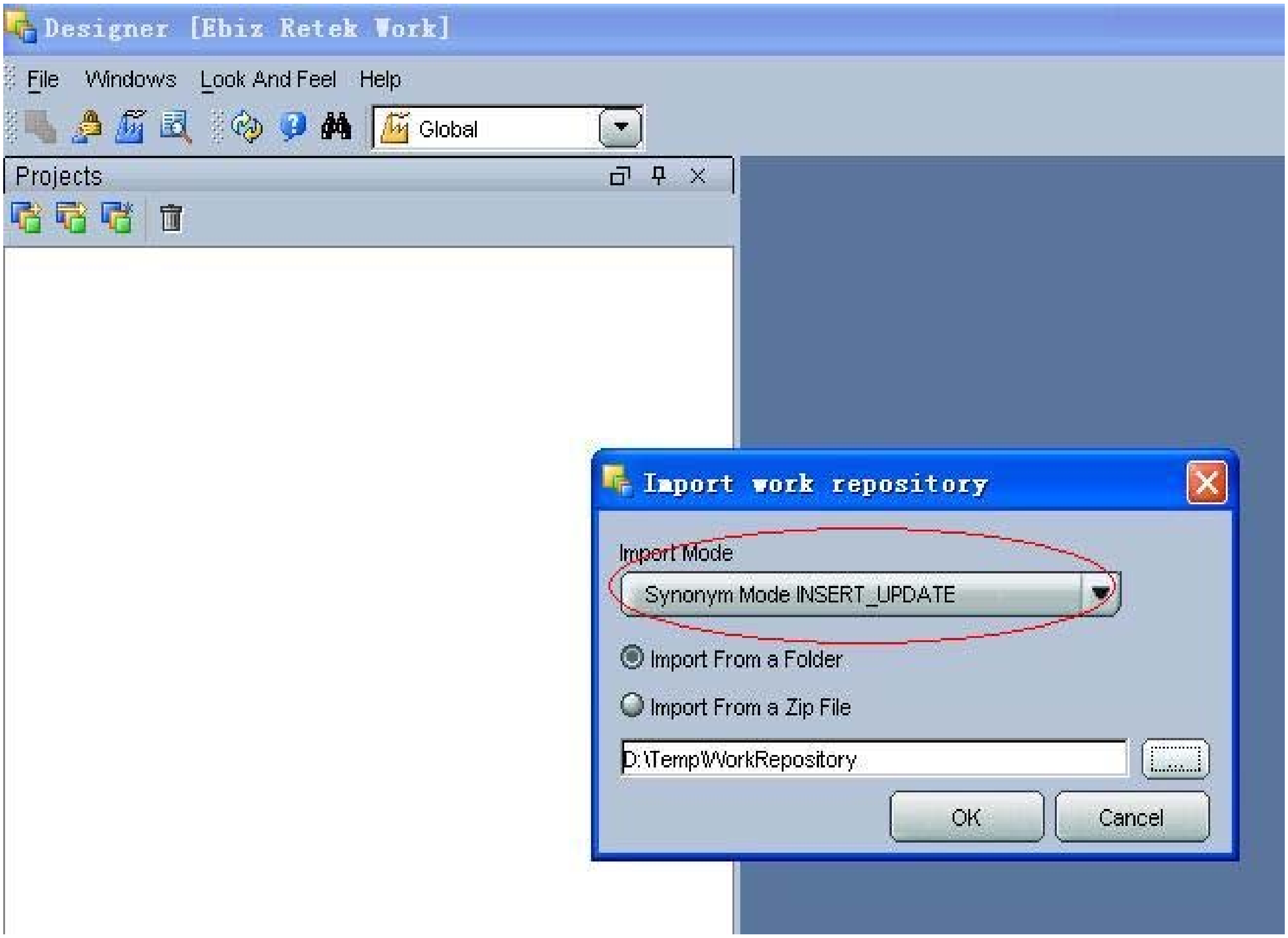
Import Work Repository

In the designer, click File->Import->Work Repository.

The work repository files are located at

\vobs\aia\PIPS\Core\DIS\RetailToEbizFin\WorkRepository

Always choose Synonym Mode INSERT_UPDATE.



Import Mode: Synonym Mode INSERT_UPDATE

● Import From a Folder

○ Import From a Zip File

D:\TempWorkRepository

Part II: Set up client side configuration and DVM files

In the computer where ODI is installed, create AIA Home directory, for example, C:\AIA_HOME\config.

Copy \vobs\aia\Infrastructure\SeedData\AIAConfigurationProperties.xml from the clear case directory to the directory you created above.

Find module RetailToEbiz_AccountingEntry and

RetailToEbiz_InvoiceEntry, change the email server and addresses to the appropriate values.

Create DVM directory, e.g. C:\DVM.

From the clear case directory, find and copy the following DVM files to the directory you created above:

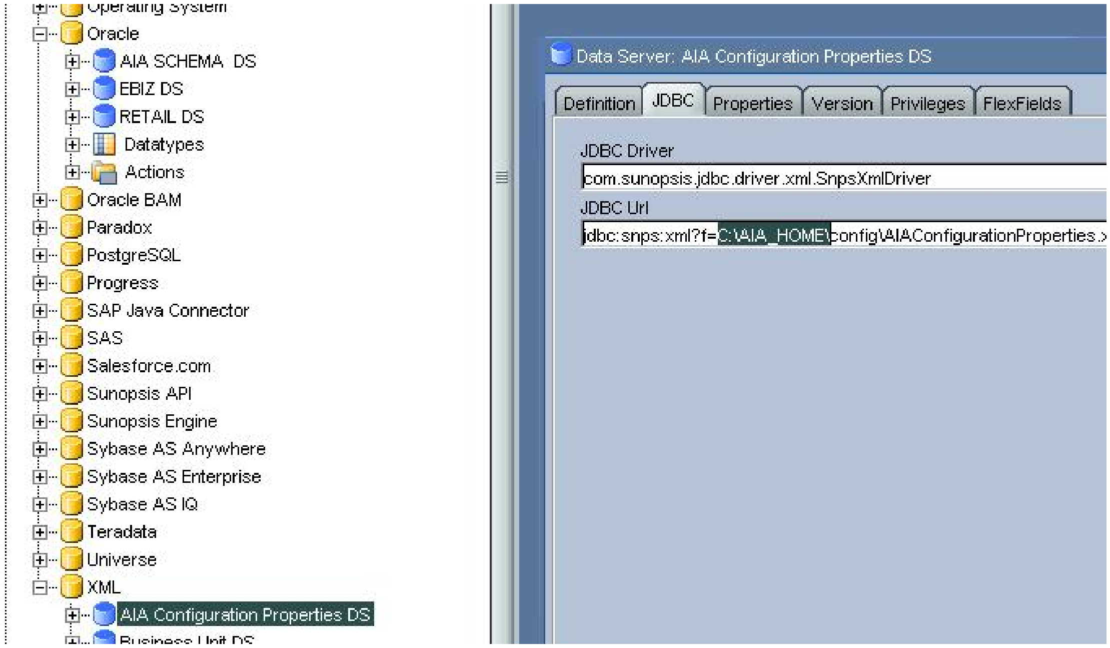
BUSINESS_UNIT.xml

CURRENCY_CODE.xml

PAYABLEINVOICE_PAYABLEINVOICELINETAXVATCODE.xml

Enter appropriate values to the DVM files.

Note: You need to check the Topology Manager -> Physical Architecture -> XML data sources to makes sure the directories are as same as the ones you created.



JDBC Driver

com.sunopsis.jdbc.driver.xml.SnpsXmlDriver

JDBC Url

jdbc:snps:xml?f=C:\AIA_HOME\config\AIAConfigurationProperties.x

Part III: Set up SOA server side XREF data

You can find the XREF files in clear case directory \vobs\aia\PIPS\Core\SeedData\XREF

Below are the XREF seed data you need to set up:

Population purpose:

- ACCOUNTINGENTRY_ACCOUNTINGENTRYID
- PAYABLEINVOICE_PAYABLEINVOICEID

Lookup purpose:

- SUPPLIERPARTY_ID
- SUPPLIERPARTY_LOCATION_ID
- ORGANIZATION_ID
- PAYMENTTERM_ID

For how to set up the XREF data, please see appendix

Part IV: Set up Ebiz source data

Login to Ebiz self-service, navigate to General Ledger responsibility, launch Categories form, enter the following categories:

Category	Key	Desc
RETEK	RETEK	RETEK
RMS	RMS	Retail RMS
Retail Invoices	Retail Invoices	Retail Invoices

Launch Sources form, enter the following sources:

Source	Key	Desc
RMS	RMS	Retail RMS
Writeoffs	Writeoffs	Retail Writeoffs
Prepayments	Prepayments	Retail Prepayments
Manual Payments	Manual Payments	Retail Manual Payments

Navigate to the Payables responsibility, in the Lookups menu, launch Payables form.

Enter the following condition and search:

Type: SOURCE Application: Payables

In the result list, add the following line and save:

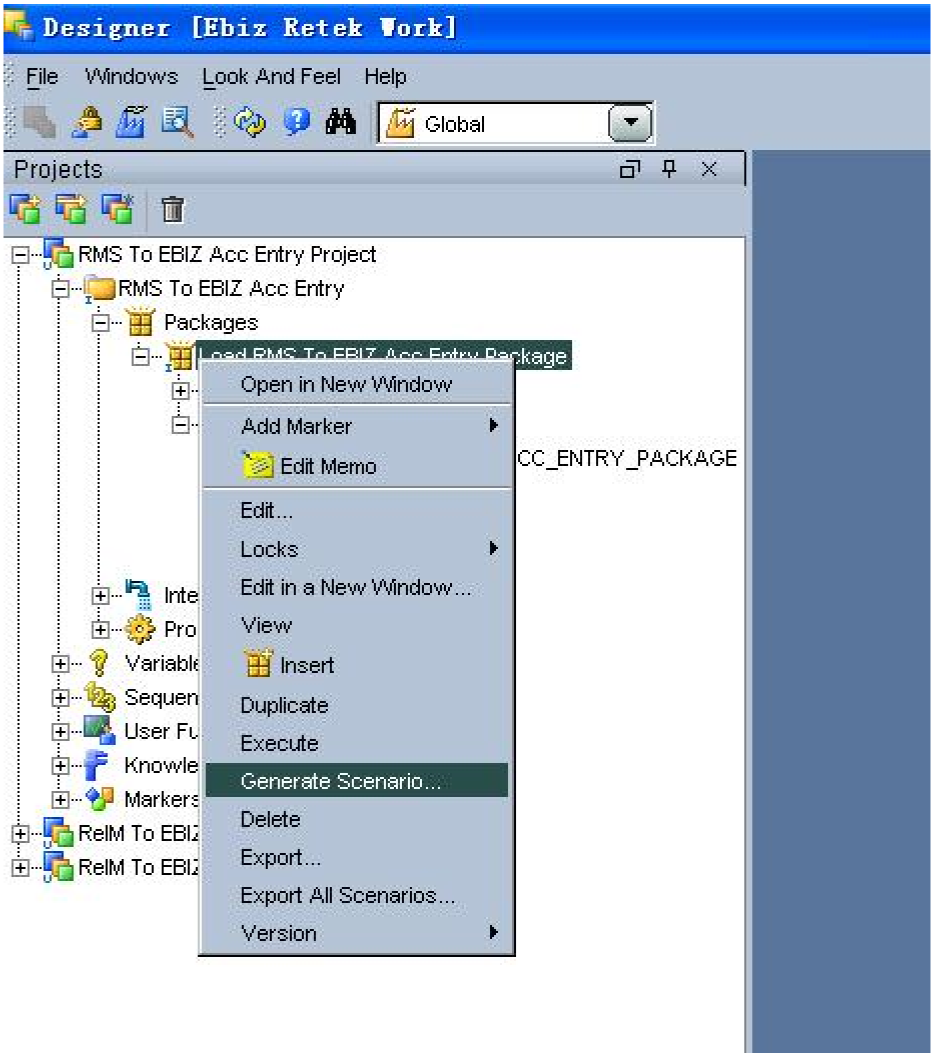
Code	Meaning	Desc
REIM	Retek ReIM	Retek ReIM

Part VI: Scenario test

In this part, you need to do similar things to the three packages in this PIP. In this guide we only use package “Load RMS To EBIZ Acc Entry Package” for example.

Generate scenario

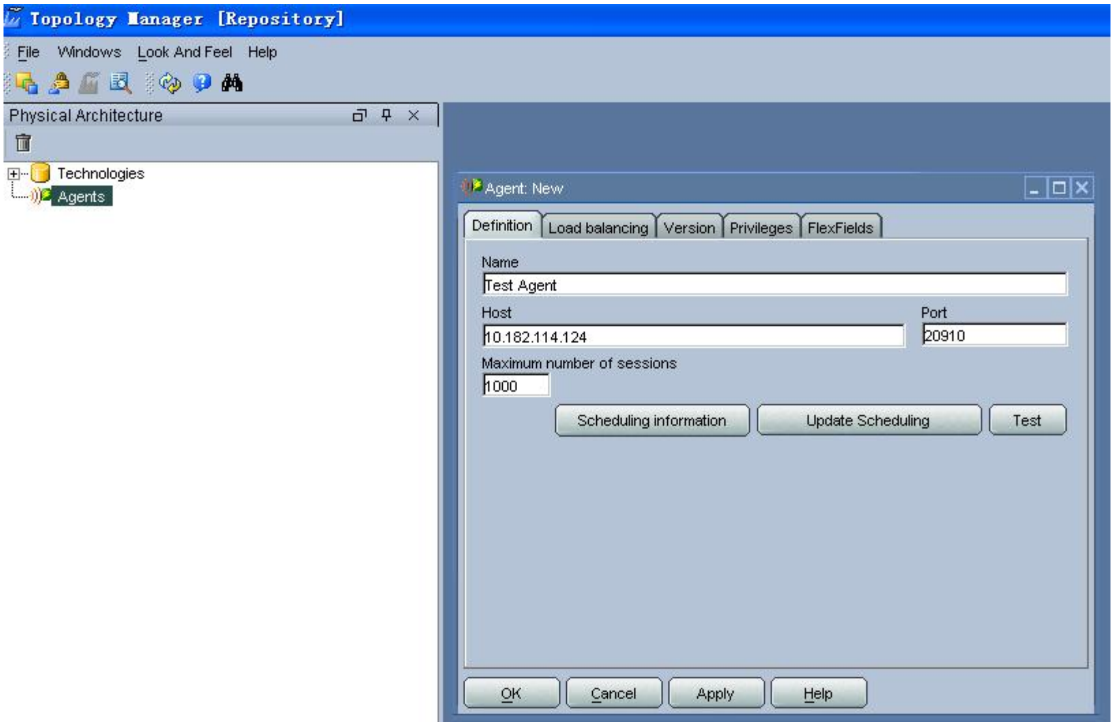
In ODI designer, navigate to the package and generate scenario, use the default values as the input.



Create physical and logical agent

This is done in the Topology Manager.

Let the port default to 20910, assign the host to the machine you are running on, name it whatever you want but remember the name.



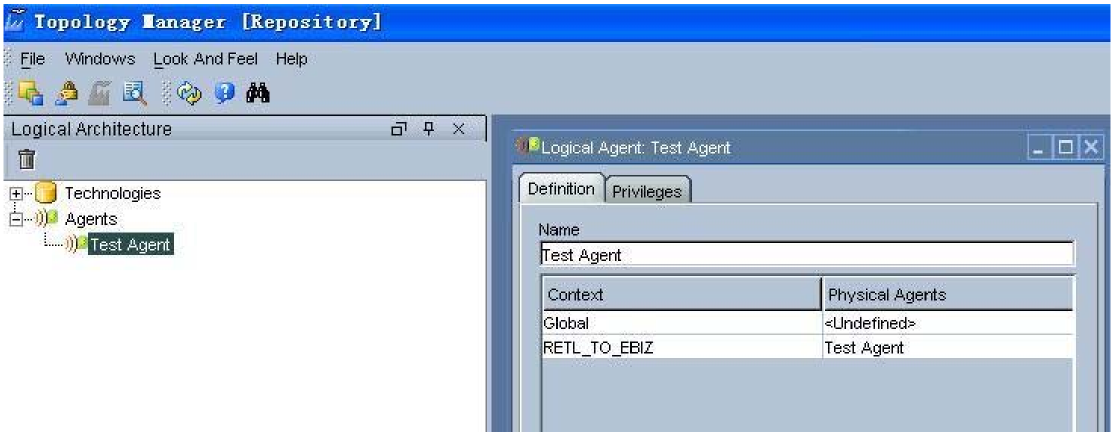
Name
Test Agent

Host
10.182.114.124

Port
20910

Maximum number of sessions
1000

Create a logical agent in Logical Architecture and assign the newly created physical agent to the context RETL_TO_EBZ.



Name
Test Agent

Context	Physical Agents
Global	<Undefined>
RETL_TO_EBIZ	Test Agent

Run agent scheduler

Modify odiparams.bat file, set the following lines to connect them to the work repository, e.g.

```
set ODI_SECU_DRIVER=oracle.jdbc.driver.OracleDriver
set ODI_SECU_URL=jdbc:oracle:thin:@ sdc60007sems.us.oracle.com:1605:retebsdv
set ODI_SECU_USER=snpm
set ODI_SECU_ENCODED_PASS=c6yHJ,CVjFSxK5,QqQHeALMx
set ODI_SECU_WORK_REP=EBIZ_RETEK_WORK
set ODI_USER=SUPERVISOR
set ODI_ENCODED_PASS=e3yaf23LwyAZNRyjpEWe.foc
```

You can use the following command to generate the encoded password:

```
agent encode <password>
```

Go to a CMD prompt and type

```
agentscheduler "-name=<physical agent name>" "-v=5"
```

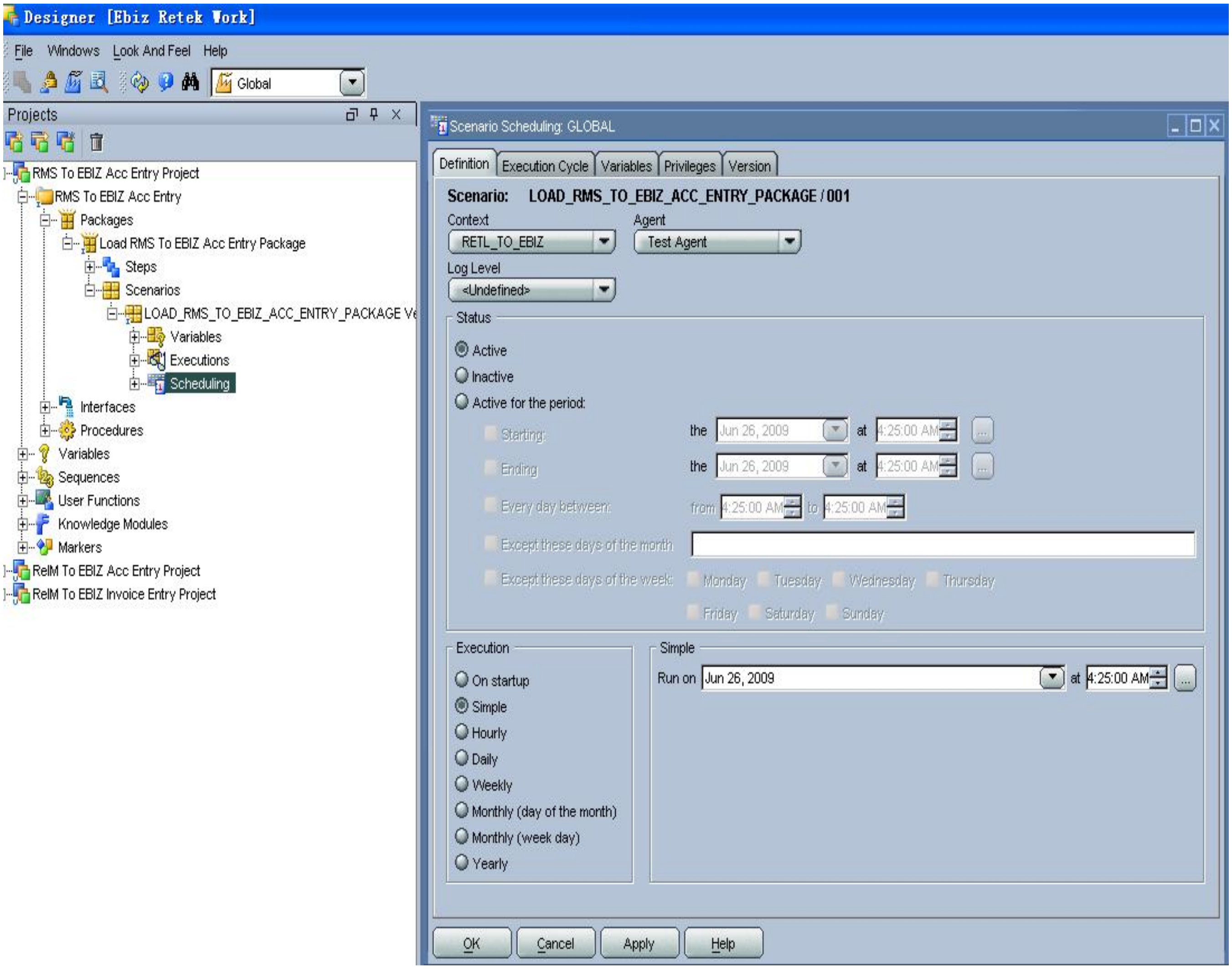
For example,

```
agentscheduler "-name=Test Agent" "-v=5"
```

You should get a message that the agent is started successfully.

Generate scheduler

Go to Designer and find the scenario you generated. Right click and insert Scheduling.


Designer [Ebiz Retek Work]
File Windows Look And Feel Help
Global
Projects
RMS To EBIZ Acc Entry Project
RMS To EBIZ Acc Entry
Packages
Load RMS To EBIZ Acc Entry Package
Steps
Scenarios
LOAD_RMS_TO_EBIZ_ACC_ENTRY_PACKAGE Ve
Variables
Executions
Scheduling
Interfaces
Procedures
Variables
Sequences
User Functions
Knowledge Modules
Markers
RelM To EBIZ Acc Entry Project
RelM To EBIZ Invoice Entry Project
Scenario Scheduling: GLOBAL
Definition
Execution Cycle
Variables
Privileges
Version
OK
Cancel
Apply
Help

Scenario: LOAD_RMS_TO_EBIZ_ACC_ENTRY_PACKAGE / 001

Context: RETL_TO_EBIZ

Agent: Test Agent

Log Level: <Undefined>

Status

● Active

○ Inactive

○ Active for the period:

□ Starting: the Jun 26, 2009 at 4:25:00 AM

□ Ending: the Jun 26, 2009 at 4:25:00 AM

□ Every day between: from 4:25:00 AM to 4:25:00 AM

□ Except these days of the month

□ Except these days of the week: □ Monday □ Tuesday □ Wednesday □ Thursday □ Friday □ Saturday □ Sunday

Execution

○ On startup

● Simple

○ Hourly

○ Daily

○ Weekly

○ Monthly (day of the month)

○ Monthly (week day)

○ Yearly

Simple

Run on Jun 26, 2009 at 4:25:00 AM

Update schedule

Go back to Topology and find your Physical Agent and Update Schedule.

Check

Go relax and wait for your agent to run at your preset time. When it runs, you can check it in Operator. You can also check the data in Ebiz side interface tables.

Run concurrent import program

When running the import program, you need to enter the source name you set up in previous steps.

When concurrent program is completed, you can check the data in SOA XREF table and Ebiz transaction tables.

New Customers to Oracle Retail to Ebiz in 2.5 Initial Load for Cross References (Post Install Directions)

Insert records into Cross-Reference table using xref utilities

To populate cross-references: (The example used below is for ORGANIZATION _ID, but can be used for any other xref table initial loads. Substitute your xref table name, column names and values.

1. Create an .xml file using the following sample. Update the XML file where the font is **BOLD** with the appropriate values:

   column name - define the columns in the table that you want to populate

   cell colName – this creates the rows and defines the values in that row for the columns

Note: the column “COMMON” is not needed in the xml file because the tool wil generate it

Sample 1 .xml: organization.xml

```
<xref xmlns="http://xmlns.oracle.com/xref">
<table name="ORGANIZATION_ID">
<columns>
 <column name="RETL_01"/>
 <column name="EBIZ_01"/>
</columns>
<rows>
 <row>
  <cell colName="RETL_01">333333333333333</cell>
  <cell colName="EBIZ_01">US001</cell>
 </row>
 <row>
  <cell colName="RETL_01">444444444444444</cell>
  <cell colName="EBIZ_01">US002</cell>
 </row>
 <row>
  <cell colName="RETL_01">555555555555555</cell>
  <cell colName="EBIZ_01">CAN01</cell>
 </row>
```

```
</rows>
</table>
</xref>
```

Sample 2 .xml: PaymentTerm_ID.xml

```
<xref xmlns="http://xmlns.oracle.com/xref">
<table name="PAYMENTTERM_ID">
<columns>
 <column name="RETL_01"/>
 <column name="EBIZ_01"/>
</columns>
<rows>
 <row>
  <cell colName="RETL_01">750</cell>
  <cell colName="EBIZ_01">30DayPay</cell>
 </row>
 <row>
  <cell colName="RETL_01">751</cell>
  <cell colName="EBIZ_01">1202</cell>
 </row>
 <row>
  <cell colName="RETL_01">752</cell>
  <cell colName="EBIZ_01">1203</cell>
 </row>
</rows>
</table>
</xref>
```

Save the xml file to a temp directory on the SOA server in the bin directory (for your particular server)

```
 $SOA_HOME/integration/esb/bin
For example:
$SOA_HOME = /slot/ems1891/oracle/product/10.1.3.1/OracleAS_1
```

NOTE: Before running the commands below, make sure that the Xref table and the columns you need to populate exist on your server. (see section **9**)

2. Make sure that you have set up your environment variables for the following, prior to running the xref tool.

   OC4J_USERNAME

   OC4J_PASSWORD

3. Run the xrefimport tool to import the cross-references by following the instructions below
4. Use Xterm or any other protocol to connect to the SOA Server.

   Xterm location: http://ebiztta.oraclecorp.com/cgibin/secure/ttawlogin.cgi/?action=start&pg=index2.html

5. On the command line, type the commands in the example below. Substitute your server and port

    Bold: commands

    Bold Italics: values in the command that needs to be substituted with the appropriate values

    Example:

```
Log in: ssh -l ora1891 adc60117fems.us.oracle.com  (ssh –l is the
lower case letter l, not numeric 1)
```

For the login, substitute host and user name with the appropriate values for your server

You might be asked the following questions:

“Are you sure you want to continue connecting (yes/no)?” say “**yes”**

At the prompt for password type in **“welcome”** (without the quotes)

You will get the following prompt:

```
-bash-3.00$  (3.00 might be different)
```

6. At the bash prompt change directories as follows:

```
-bash-3.00$ cd product/10.1.3.1/OracleAS_1/integration/esb/bin
 You will get the -bash prompt again
```

7. If you first need to add the xref table and columns or just the columns on the server, type: (If this is already done, move to number 10)

```
-bash-3.00$ chmod +x xreftool.sh
-bash-3.00$ ./xreftool.sh -shell
-bash-3.00$ export OC4J_USERNAME=oc4jadmin
-bash-3.00$ export OC4J_PASSWORD=welcome1
-bash-3.00$ ./xreftool.sh -shell
```

The prompt should change to >

addTable ORGANIZATION_ID

addColumns ORGANIZATION_ID EBIZ_01,COMMON,RETL_01

```
   Organization_Id xref table has just been added with the columns
needed for the initial load.
```

8. Type in the following commands to import the data in the .xml file to the xref table. This will be using the xrefimport tool. After each commnad, you will be returned to the –bash prompt

```
-bash-3.00$ chmod +x xrefimport.sh
-bash-3.00$ ./xrefimport.sh -shell
```

9. Set the following environment variables with the correct values, using the script below:

```
DB_USER
DB_PASSWORD
OC4J_USERNAME
OC4J_PASSWORD
DB_URL
```

```
Example:
-bash-3.00$ export DB_USER=aia
-bash-3.00$ export DB_PASSWORD=aia
-bash-3.00$ export OC4J_USERNAME=oc4jadmin
-bash-3.00$ export OC4J_PASSWORD=welcome1

(the next three lines need to go in one line)
-bash-3.00$ export
      DB_URL="jdbc:oracle:thin:@adc60117fems.us.oracle.
            com:1612:finandb"
```

(substitute the following with values for your server)

```
db hostname - adc60117fems.us.oracle
jdbc port - 1612
sid - finandb
```

10. Import the data using the command below:

```
   -bash-3.00$ xrefimport.sh -file \TEMP/organization_id.xml  -mode
         ignore -generate COMMON
Parameters:
   -file = your xml file name with the directory you saved it in
(your file should be in a folder in the bin directory) Since you are
already in the bin directory, you don’t need to change dir Required)
   -mode = defines how to handle conflicts with existing data in the
xref table. There are two possibilities for mode (Optional)
   -mode ignore = existing records in the xref table will be ignored
or left as is.  New data from the xml file will be appended to the
existing data if different.  If the data is a duplicate for a row, a
new record will not be written.
   -mode overwrite = if there is a conflict with the data, the
   existing data in question will be overwritten with the new data
from the XML file
    -generate COMMON = this tells the tool to generate a common
record for each set of rows with an auto-generated guid. It links
the records together by row number (Optional)
```

Validating Cross-References

To validate cross-references:

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

   Use the following query:

```
The query below will retrieve all records in the ORGANIZATION_ID
TABLE
select * FROM xref_data where xref_table_name='ORGANIZATION_ID'


or
```

```
The query below will retrieve the three associated records for the
 org_id you selected
 select * from xref_Data
 where row_number in (select row_number from xref_Data
 where value = 'CAN01')
```

Replace the value for the organizations you selected. (The number of operating units depends on your setup.)

3. Validate that for each row, three records were written.

   RETL_01, EBIZ_01 and COMMON

   The three records should be linked to each other with a common row number value

For more information about creating schedules and scenarios, see *Oracle Data Integrator User Guide* and *Oracle Data Integrator Installation Guide*. These documents are available in <odi install location?>\oracledi\doc subdirectory.

Running the Process Integration for Accounting Entries

This process runs based on the schedules created for it. The integration between Oracle Retail and Oracle GL can be scheduled to occur automatically.

Note: You can start or stop the scheduler agent whenever you want, which enables you to control the integration process.

For more information about starting and stopping the scheduler agent, see *Oracle Data Integrator User Guide*, “Launching a Scheduler Agent” and “Stopping an Agent.” This document is available in <odi install location?>\oracledi\doc subdirectory

Data Requirements

The accounting entry and invoice ODI integration flows are run.

For more information, see Chapter 4: Process Integration for Accounting Entries and Chapter 5: Process Integration for Retail Merchandise Procure to Pay.

Chapter 6: Implementing Oracle Retail Merchandise Integration Pack for E-Business Suite Financials

This chapter discusses how to:

- Set up Oracle Retail applications
- Set up Oracle General Ledger
- Set up Oracle Payables
- Set up Oracle Domain Value Maps
- Manually synchronize freight terms
- Manually synchronize currency rate types
- Manually synchronize currency codes
- Manually synchronize chart of accounts values
- Manually synchronize financial calendar
- Map business and organization units
- Work with cross-references
- Work with domain value maps (DVMs)
- Handle errors
- Set configuration properties

Oracle Retail applications, Oracle General Ledger (GL), and Oracle Payables must be set up for the Oracle Retail to Oracle process integration pack (PIP) to work properly. This chapter describes these setups in detail.

Setting Up Oracle Retail Applications

Before integration, you must set up Oracle Retail Merchandising System (RMS), Oracle Retail Invoice Matching (ReIM), and oracle Retail Sales Audit.

For more information, see the *ReIM Operations Guide*, “PeopleSoft Enterprise Financials Integration” and the *RMS Operations Guide, Volume 3*, “AIA Financials Integration” for setting up the Oracle Retail applications

Setting Up Oracle GL

All setup steps described in implementation guide.

To set up Oracle GL, perform these actions:

- Activate service operations, queue, and handlers
- Activate or create routings
- Set up end points (connector properties) for provided services
- Setup recurring run controls for delay publish of effective dated objects
- Activate validation service operations
- Set up cross-reference tables
- Synchronize currency exchange rate

For more information, see Oracle General Ledger Implementation Guide.

Setting Up Oracle Payables

All setup steps described in implementation guide.

To set up Oracle Payables, perform these actions:

- Activate service operations, queue, and handlers
- Activate or create routings
- Set up end points (connector properties) for provided services
- Set up recurring run controls for delay publish of effective dated objects
- Activate validation service operations
- Set up cross-reference tables
- Synchronize payment terms and vendors

For more information, see Oracle Payables Implementation Guide.

Setting Up Cross-References for Oracle Retail IDs and Oracle Entities

Before running the supplier flow, you must load the initial ORGANIZATION_ID cross-reference table. Because no automated process for this synchronization is available, you need to load this table manually.

This section discusses:

- Populating cross-references
- Validating cross-references
- Viewing cross-reference data

Populating Cross-References

To populate the cross-references:

1. Create an xml file using the following sample.

Note: This script is only for initial load. It also enables customers to add new business unit required for their organizations. The process is same for both. It does not update the cross-reference table. There is no update capability.

Sample Organization.xml:

```
<xref xmlns="http://xmlns.oracle.com/xref">
<table name="ORGANIZATION_ID">
<columns>
   <column name="RETL_01"/>
   <column name="EBIZ_01"/>
</columns>
<rows>
   <row>
         <cell colName="RETL_01">333333333333333</cell>
         <cell colName="EBIZ_01">US001</cell>
   </row>
   <row>
         <cell colName="RETL_01">444444444444444</cell>
         <cell colName="EBIZ_01">US002</cell>
   </row>
   <row>
         <cell colName="RETL_01">555555555555555</cell>
         <cell colName="EBIZ_01">CAN01</cell>
   </row>
</rows>
</table>
</xref>
```

Sample 2 .xml: PaymentTerm_ID.xml

```
<xref xmlns="http://xmlns.oracle.com/xref">

<table name="*PAYMENTTERM_ID*">

<columns>

<column name="*RETL_01*"/>

<column name="*EBIZ_01*"/>

</columns>

<rows>

<row>

 <cell colName="*RETL_01*">*750*</cell>
 <cell colName="*EBIZ_01*">*10001*</cell>
</row>
<row>
 <cell colName="*RETL_01*">*751*</cell>
 <cell colName="*EBIZ_01*">*10002*</cell>
</row>
<row>
 <cell colName="*RETL_01*">*752*</cell>
 <cell colName="*EBIZ_01*">*10003*</cell>
</row>
</rows>
</table>
</xref>
```

Note: Substitute anything in bold and variables with the appropriate values. The sample uses only three row sets, create as many as needed. (Variables are preceded with a $) The data for this file needs to be coordinated between PeopleSoft and Retail. No common column exists because this is auto generated by the tool.

Save the xml file to a temp directory on the SOA server in the bin directory (for your particular server).

$SOA_HOME/integration/esb/bin

For example: $SOA_HOME = /slot/ems1891/oracle/product/10.1.3.1/OracleAS_1

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. Prior to run the xrefimport utility, set up the environment variables for the following:

   a. OC4J_USERNAME

   b. OC4J_PASSWORD

3. Run the xrefimport utility to import the cross-reference data using the steps 4 to 9.

4. Use Xterm or any other protocol to connect to the SOA server.

5. On the command line, enter the following commands:

   Substitute your server and port. All commands are in bold.

For example:

Log in: **ssh -l $USER_NAME $HOST_NAME** (ssh –l is the lower case letter l, not numeric 1)

You might be asked the following questions:

“Are you sure you want to continue connecting (yes/no)?”
select “**yes”**

At the prompt for password, enter **$LOGIN_PASSWORD**

You will get this prompt: -bash-3.00$ (3.00 might be different)

6. At the bash prompt, change directories to the directory when the “bin folder” resides under your Oracle home:

For example:

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

You will get the -bash prompt again.

7. Enter the following commands to import the data that is in the .xml file for the xref table.

This uses the xrefimport tool. After each command, you will be returned to the –bash prompt

a. -bash-3.00$ chmod +x xrefimport.sh

b. -bash-3.00$ ./xrefimport.sh –shell

(This is an optional command. It brings up the environment variables that need to be set)

c. DB_USER

d. DB_PASSWORD

e. OC4J_USERNAME

f. OC4J_PASSWORD

g. DB_URL

8. Set the environment variables mentioned in the step 7 with the correct values using the following script:

Example:

```
-bash-3.00$ export DB_USER=$db_user

-bash-3.00$ export DB_PASSWORD=$db_password

-bash-3.00$ export OC4J_USERNAME=$oc4j_username

-bash-3.00$ export OC4J_PASSWORD=$oc4j_password

-bash-3.00$ export DB_URL="jdbc:oracle:thin:@$HOST_NAME: $JDBC_PORT:$SID"
```

9. Import the data using the following command:

   -bash-3.00$ **./xrefimport.sh -file \TEMP/$filename.xml -generate COMMON**

   Parameters:

   -**file =** your xml file name with the directory name where you saved the xml file (your file should be in a TEMP folder in the bin directory).

   -generate COMMON= this tells the tool to generate a common record for each set of rows with an auto-generated guid. It links the records together by row number (optional)

10. To view the data in the xref table in a document, use the export utility while in the command line. This utility exports the data from the xref table and creates a document with the specified name and location. If you have access to the database, you can copy the data into an excel spreadsheet.

For more information, see Viewing Cross-Reference Data.

Validating Cross-References

Sample To validate the cross-references:

1. Log in to your database.
2. Query the Table ORGANIZATION_ID to confirm that every organization/business unit used in the XML files has three records.

   Use the following query:

   ```
   select * FROM xref_data where xref_table_name='ORGANIZATION_ID'
   ```
   This query retrieves all records in the ORGANIZATION_ID TABLE.

   or

   ```
   select *from xref_data where xref_table_name='ORGANIZATION_ID' and
   xref_column_name='EBIZ_01'
   ```
   This query can be modified to change the column name to RETA_01 and COMMON.

   Or

   ```
   select * from xref_Data
   ```
   where row_number in (select row_number from xref_Data
   where value = '**CAN01**')

   The query retrieves the three associated records for the org_id you selected.

   Replace the value for the business unit you selected. (The number of business units depends on your setup.)

3. Validate each row for which the following three records are written:

   RETL_01, EBIZ_01 and COMMON

The three records should be linked to each other with a common row number value.

The data from SQL can be exported or copied to an excel spreadsheet. You can sort this data by row number, and so on.

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

Viewing Cross-Reference Data

Sample to view the cross-reference data

The data in the cross-reference table needs to be synched with the Business Unit DVM in E-Business Suite. Because the COMMON is an auto generated guid, and not known to the E-Business Suite customer, data visibility is essential. This data need to be entered in the E-Business Suite DVM.

1. Exporting To File: After the records have been written to the cross-reference table, the data can be viewed easily by exporting it to another document.

   Run the xrefexport utility to export the cross-reference data using these instructions:

   a. Use Xterm or any other protocol to connect to the SOA Server.

   b. On the command line, type the following commands. Substitute with your server and port. Commands are bold.

      For example:

      Log in: ssh -l $USER_NAME $HOST_NAME

      (ssh –l is the lower case letter l, not numeric 1)

      You might be asked the following questions:

      “Are you sure you want to continue connecting (yes/no)?” say “yes”

      At the prompt for password, enter $LOGIN_PASSWORD

      You will get the following prompt:

      -bash-3.00$ (3.00 might be different)

   c. At the bash prompt, change directories to the directory when the “bin folder” resides under your Oracle home:

      For example:

      -bash-3.00$ cd $OracleHome/integration/esb/bin

      You will get the -bash prompt again

   d. Enter the following commands to import the data that is in the xml file for the xref table. This will be using the xrefimport tool.

After each command, you will be returned to the –bash prompt

```
-bash-3.00$ chmod +x xrefexport.sh
```

-bash-3.00$ ./xrefexport.sh –shell (this command is optional, it will bring up the environment variables that needs to be set)

DB_USER

DB_PASSWORD

OC4J_USERNAME

OC4J_PASSWORD

DB_URL

e. Set the mentioned environment variables with the correct values, using this script:

   For example:

```
-bash-3.00$ export DB_USER=$db_user

-bash-3.00$ export DB_PASSWORD=$db_password

-bash-3.00$ export OC4J_USERNAME=$oc4j_username

-bash-3.00$ export OC4J_PASSWORD=$oc4j_password

-bash-3.00$ export DB_URL="jdbc:oracle:thin:@$HOST_NAME: $JDBC_PORT:$SID"
```

f. Export the data using this command:

```
-bash-3.00$ ./xrefexport.sh  -file \TEMP/$filename.xml  -table ORGANIZATION_ID
```

   Parameters:

   -file = the tool will create a file for you with the name you give it. (This file can also have a .txt extension)

   -table = this tells the tool which xref table to use for the export

2. Copy and paste from the SQL Worksheet:

   If you have access to the database, the data can also be copied into an excel spreadsheet. From a sql worksheet:

   a. Query on the table

   b. select * FROM xref_data where xref_table_name='ORGANIZATION_ID'

   c. right click on the displayed data, select all/copy and paste into an excel spreadsheet

Setting Up Oracle Domain Value Maps

Set up these Oracle domain value maps (DVMs):

Name	Description
BUSINESS UNIT	Business Unit Mapping
CONVERSIONTYPECODE	Rate type mapping
CURRENCY_CODE	Currency code mapping
CUURENCYEXHANGE CONVERSIONTYPECODE	Currency Exchange Conversion Type Mapping
LANGUAGE_CODE	Language Code DVM
STATE	State Code DVM
ADDRESS COUNRTY ID	Country Code DVM
SUPPIER PARTY_ADDRESSTYPE	Supplier Address Type DVM
SUPPLIERPARTY_STATUSCODE	Supplier Type DVM

Manual Synchronization of Freight Terms

Freight is the transportation charge paid by retailers when they receive goods from a supplier. *Freight term* is an agreement between a retailer and a supplier regarding the type and payment of freight.

Because the volume of freight terms is low, they are maintained and synchronized manually in Oracle Payables and RMS

Manual Synchronization of Currency Exchange Rate Types

Oracle Retail allows for multiple currency exchange rate types (such as operational, consolidated, letter of credit or bank, purchase order, customs entry, and transportation) but uses only one type for the default processing—either Operational or Consolidation.

The Oracle Financials system supports any number of currency exchange rate types and allows for multiple currency exchange rate types. Because Oracle Retail uses only one currency exchange rate for the default processing, one exchange rate is selected. The selected currency exchange rate is used by all of the Oracle Retail products that integrate with E-Business Suite.

The E-Business Suite system can have separate currency exchange rate types that are not integrated with Oracle Retail. The relationship between Oracle Retail and Oracle rate types is mapped in a Domain Value Mapping (DVM) table.

When the selected currency exchange rate type is manually set up in RMS, RMS uses the Currency Exchange Type Mapping window to map the external exchange type sent by the E-Business Suite system. This currency exchange rate type is used by the Oracle Retail applications for all transactions.

Because the volume of currency exchange rate types is low, they are maintained and synchronized manually in Oracle GL and RMS. These are not synchronized automatically. This synchronization is required for the synchronization of currency exchange rates between Oracle GL and RMS.

For more information, see the *Oracle General Ledger implementation guide.*

Manual Synchronization of Currency Codes

Oracle Retail and Oracle GL synchronize the currency codes manually. The currency codes mapping between Oracle GL and Oracle Retail is required for the sales audit and stock ledger transactions.

For more information, see the *Oracle General Ledger implementation guide.*

Manual Synchronization of Chart of Accounts Values

A chart of accounts is a listing of all the accounts in the general ledger. You can use a chart of accounts to view specific information about a transaction.

Oracle Retail uses the chart of accounts as reference information to create accounting entries for both sales and inventory transactions. Oracle General Ledger (GL) is the system of record for this information. Oracle Retail also stores and uses the GL accounts.

In E-Business Suite, the chart of accounts implies all the accounting segments and values such as chart of accounts, business unit, and setID. A chart of accounts includes account, department, operating unit, and other segments. The attributes under each segment can differ from those of other segmetn. For example, Account can have Account Number, Description, Account Type, and VAT Flag, and Department can have Department number, Description, and Manager.

Chart of accounts is manually synchronized from Oracle Payables to Oracle Retail. It is not synchronized automatically.

The chart of accounts is validated based on the currently chart of account. The most current active Chart of Account value row needs to be manually synchronized from Oracle GL to Oracle Retail.

The E-Business Suite system can also create future dated chart of accounts. These future dated charts of accounts need to be re-validated against E-Business Suite. Also, chart of accounts can be end-dated or disable. The chart of account will need to be manually re-validated.

After the chart of accounts values have been set up in both systems, a validation occurs for each Oracle Retail transaction ready to be transmitted to Ebusiness Suite system

Manual Synchronization of Financial Calendar

The calendars are manually maintained in Oracle Retail and E-Business Suite; they are not automatically synchronized.

For more information, see the *Oracle General Ledger implementation guide.*

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 Oracle Retail to Oracle Financials PIP cross-references:

Cross Reference Table Name	Column Name	Description	Usage
CURRENCYEXCHANGE_ID	COMMON	Oracle GUID	Populate during the currency exchange integration.
	RETL_01	Concatenation of Retail FromCurrencyCode:To CurrencyCode:ConversionType: ConversionDate	
	EBIZ_01	Oracle GUID	
PAYABLEINVOICE_PAYABLEINVOICEID	COMMON	Same GUID as PSFT_01	Populated by the Oracle Data Integrator (ODI) invoice integration.
	RETL_01	Reference ID from Oracle Retail	
	EBIZ_01	A GUID generated by the native database	
PAYMENTTERM_ID (manual sync)	COMMON	Generated GUID	Populate during the payment term integration.
	RETL_01	Oracle Retail ID	
	EBIZ_01	Oracle GUID	

Cross Reference Table Name	Column Name	Description	Usage
SUPPLIERPARTY_ID	COMMON	Generated GUID	Populate during the supplier integration.
	RETL_01	Oracle Retail ID	
	EBIZ_01	Oracle GUID	
SUPPLIERPARTY_LOCATION_ID	COMMON	Generated GUID	Populate during the supplier integration.
	RETL_01	Oracle Retail ID	
	EBIZT_01	Oracle GUID	

Working with DVMs

DVMs are a standard feature of the Oracle Service Orientation Architecture (SOA) suite. They are tables containing mapping between related information in the participating applications. They enable you to equate lookup codes and other static values across applications, for example, FOOT and FT or US and USA. These DVM tables are maintained in the AIA layer. The AIA layer uses these DVM tables in transforming the messages from one system in the expected format of the other system.

DVM types are seeded for the Oracle Retail to Oracle Financials 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.

Note: For any DVM that is not pre-seeded, the first row has description information such as Enter Org Unit here. Remove this first row because it causes issues in the Oracle Data Integrator flow.

This table lists the DVMs for the Oracle Retail to Oracle Financials PIP:

DVM Type	DVM Column Name	Comments
BUSINESS_UNIT	EBIZ_01, COMMON, RETL_01	This maps Oracle Retail's Set Of Books ID to Oracle GL business unit.

DVM Type	DVM Column Name	Comments
CHARTOFACCOUNTS_ACCOUNTSTATUS	EBIZ_01, COMMON, RETL_01	This is a hard coded DVM and should not be changed. This maps the Oracle and EBO value of “true” or “false” to Oracle Retail’s value of “valid” or “invalid”.
ADDRESS_COUNTRYID	COMMON, RETL_01, PSFT_01	This maps the country codes between Oracle Retail and Oracle system.
CURRENCYEXCHANGE_CONVERSIONTYPECODE (CURRENCYEXCHANGE95CONVERSIONYPECODE)	EBIZ_01, COMMON, RETAIL _01	
CURRENCY_CODE (CURRENCY95CODE)	EBIZ_01, COMMON, RETL_01	This maps the currency codes between Oracle Retail and Oracle system.
CURRENCYEXCHANGE_STATUSCODE	EBIZ_01, COMMON, RETAIL_01	Retail has not the status field, the column’s value are null.
LANGUAGE_CODE	EBIZ_01, COMMON, RETL_01	This maps the language code between Oracle Retail and E-Business Suite system. This DVM is used to determine which language to be sent the translatable fields to Oracle Retail.
ORGANIZATION_ID	EBIZ_01,Common, RETAIL _01	Common will be a GUID. XREF for Retail will be the ORGANIZATIONUNITID and value for Ebiz will be ORG_ID.
ORACLERETAILSEGMENTCOLNAMES	STG_FIF_GL_DATA, IM_FINANCIALS_STG, IM_AP_STAGE_DETAIL , IM_AP_STAGE_HEAD	This is a hard coded DVM and should not be changed. This lists the column names for all 20 segments for the tables in the RMS, REIM, and REIM Journal Entry systems. This is used for the accounting entry ODI flow.

DVM Type	DVM Column Name	Comments
STATE	EBIZ_01, COMMON, RETL_01	This maps the state codes between Oracle Retail and Oracle Payables.
SUPPLIER_CONTACT_ID	EBIZ_01, Common,OTM_01	Common will be a GUID. XREF for OTM will be the “Domain” names and value for EBIZ will be ORG_ID
SUPPLIERPARTY_ADDRESSTYPE	EBIZ_01, COMMON, RETL_01	This maps the supplier’s address type between Oracle Retail and Oracle Payables.
SUPPLIERPARTY_ID	EBIZ_01, COMMON, RETAIL_01	Common will be a GUID generated by BPEL. Xref value for Retail is Supplier/Supplier_Name. The value for Ebiz will be the VendorID.
SUPPLIERPARTY_ADDRESS_ID	EBIZ_01, COMMON, RETAIL_01	Common will be GUID, Ebiz will be Sites/LocationID. Retail’s field for Address is AddressID.
SUPPLIERPARTY_STATUSCODE	EBIZ_01, COMMON, RETL_01	This maps the supplier’s status between Oracle Retail and Oracle Payables.
SUPPLIERPARTY_LOCATION_ID	EBIZ_01, Common, RETAIL _01	Common will be GUID, Ebiz will be VENDOR_SITEID, Retail will be SupplierLocationID.

Handling Errors

For more information about the errors generated by Oracle Retail or Oracle applications, 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”

Setting Configuration Properties

Set these properties in the AIAConfigurationProperties.XML file. This file is available 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*, “Using the BSR,” Loading Oracle AIA Configuration File Updates.

Setting for the ProcessGLAccountValidationEBIZProvABCSImpl service property:

Property Name	Value/Default Values	Description
Default.SystemID	EBIZ_01	Oracle system instance code, defined in Business Service Repository (BSR), from the account validation provider.
ABCSExtension.PreProcessEBM	true/false Default = false	Controls whether to route messages to the extensibility system.
ABCSExtension.PostProcessABM	true/false Default = false	Controls whether to route messages to the extensibility system.

Setting for the SyncCurrencyExchangeListEbizReqABCSImpl service property:

Property Name	Value/Default Values	Description
Default.SystemID	EBIZ_01	E-Business Suite system instance code (defined in BSR) from which messages originate.
Routing.RouteToCAVS	true/false Default = false	Controls whether to route messages to the verification system.
ABCSExtension.PreXformABMtoEBM	true/false Default = true	Controls whether to route messages to the extensibility system.
ABCSExtension.PreProcessEBS	true/false Default = true	Controls whether to route messages to the extensibility system.
Transformation.EnableExtensions	true/false Default=false	This property should be set to true when customers want to customize the attribute mapping done in xsl.

Setting for the SyncCurrencyExchangeListRetailProvABCSImpl service property:

Property Name	Value/Default Values	Description
Default.SystemID	RETL_01	Oracle Retail system instance code (defined in BSR) from the currency exchange provider.
Routing.SyncCurrencyExchangeListRetailProvJMSProducer.RouteToCAVS	true/false Default = false	Controls whether to route messages to the verification system
Routing.SyncCurrencyExchangeListRetailProvJMSProducer.RETL_01.EndpointURI	No default value	Oracle Retail currency exchange endpoint wsdl location. This is a URL that normally resides on the SOA server.
Routing.SyncCurrencyExchangeListRetailProvJMSProducer.CAVS.EndpointURI	No default value	CAVS endpoint location to simulate a sync response for the Oracle Retail currency exchange endpoint wsdl location.
ABCSExtension.PreProcessABM	true/false Default = false	Controls whether to route messages to the extensibility system.
ABCSExtension.PreProcessEBM	true/false Default = false	Controls whether to route messages to the extensibility system.
ABCSExtension.PostProcessEBM	true/false Default = false	Controls whether to route messages to the extensibility system.
ABCSExtension.PostProcessABM	true/false Default = false	Controls whether to route messages to the extensibility system.

Setting for the SyncSupplierPartyListRetailProvABCSImpl service property:

Property Name	Value/Default Values	Description
Default.SystemID	RETL_01	Oracle Retail system instance code (defined in BSR) from the supplier party provider.
Routing.SupplierServicePortFactory.RouteToCAVS	true/false Default = false	Controls whether to route messages to the verification system.
Routing.SupplierServicePortFactory.RETL_01.EndpointURI	No default value	Oracle Retail SupplierParty web service endpoint location. This is a SOAP endpoint URL.
Routing.SupplierServicePortFactory.CAVS.EndpointURI	No default value	CAVS endpoint location to simulate a sync response for the Oracle Retail SupplierParty service endpoint wsdl location.
Retail.Language	1	The Language of the Oracle Retail System. This value corresponds with the LANGUAGE_CODE DVM table.

Property Name	Value/Default Values	Description
ABCSExtension.PreProcessABM	true/false Default = false	Controls whether to route messages to the extensibility system.
ABCSExtension.PreProcessEBM	true/false Default = false	Controls whether to route messages to the extensibility system.
ABCSExtension.PostProcessEBM	true/false Default = false	Controls whether to route messages to the extensibility system.
ABCSExtension.PostProcessABM	true/false Default = false	Controls whether to route messages to the extensibility system.

Setting for the SyncSupplierPartyListEbizlReqABCSImpl service property:

Property Name	Value/Default Values	Description
Default.SystemID	EBIZ_01	Oracle Payable system instance code (defined in BSR) from the supplier party provider.
Routing.RouteToCAVS	true/false Default = false	Controls whether to route messages to the verification system.
ABCSExtension.PreXformABMtoEBMSupplierPartyListABM	true/false Default = true	An Enterprise Business Flow can also invoke custom code during its execution. These will serve as extensibility points. Typical ABCS can have four Extension points. This property is used as an extension point before ABM is transformed to EBM. It determines invocation of service at the extension point is to be made or not depending on whether it is true or false
ABCSExtension.PreInvokeEBSSupplierPartyEBM	true/false Default = true	An Enterprise Business Flow can also invoke custom code during its execution. These will serve as extensibility points. Typical ABCS can have four Extension points. This property is used as an extension point after ABM to EBM transformation and before Invoking the EBS. It determines invocation of service at the extension point is to be made or not depending on whether it is true or false
Transformation.EnableExtensions	true/false Default = false	This property should be set to true when customers want to customize the attribute mapping done in xsl.

Setting for the ProcessGLAccountValidationRetailReqABCSImpl service property:

Property Name	Value/Default Values	Description
Sender.SystemID	RETL_01	Oracle Retail system instance code (defined in BSR) from which messages originate.
Routing.RouteToCAVS	true/false Default = false	Controls whether to route messages to the verification system.
ABCSExtension.PreProcessABM	true/false Default = false	Controls whether to route messages to the extensibility system.
ABCSExtension.PreProcessEBM	true/false Default = false	Controls whether to route messages to the extensibility system.
ABCSExtension.PostProcessEBM	true/false Default = false	Controls whether to route messages to the extensibility system.
ABCSExtension.PostProcessABM	true/false Default = false	Controls whether to route messages to the extensibility system.

Index