Oracle® Telecommunications Service Ordering

Process Guide

Release 12.1

Part No. E13460-02

April 2009

ORACLE

Oracle Telecommunications Service Ordering Process Guide, Release 12.1

Part No. E13460-02

Primary Author: Prashanti Gajjala

Contents

Send Us Your Comments

Preface

1 Introduction to Oracle TSO

Overview of Oracle Telecommunications Service Ordering 1-1

2 Oracle TSO Business Processes

Oracle TSO Business Processes Overview 2-1
Configuring New Item Instances 2-1
Reconfiguring Installed Item Instances 2-3
Disconnecting Installed Item Instances 2-5
Changing Item Instances from Oracle Installed Base 2-7
Pricing a Configuration 2-8
Placing the Order and Verifying Order Fulfillment 2-8
Integration with Oracle Telecommunications Billing Integrator 2-8

3 Oracle TSO Functionality

Overview of Key Functionality 3-1
Oracle Configurator Functionality 3-4
Oracle Installed Base Functionality 3-6
Functionality of Sales Agent-Enabled Applications and Oracle iStore 3-7
Oracle Advanced Pricing Functionality 3-19
Oracle Service Fulfillment Manager Functionality 3-19
Oracle Telecommunications Billing Integrator Functionality 3-20

4 Implementing the Oracle TSO Solution

Overview of Implementing the Oracle TSO Solution 4-1
Mandatory Dependencies for the Oracle TSO Solution 4-2
Optional Integrations 4-5
Oracle TSO Setup Sequence 4-5
Implementation Flows for Specific Functionality 4-6

5 Set Up Oracle Order Management

Overview of Set Up Oracle Order Management Chapter 5-1
Map Actions and Line Types 5-2
Enable Recurring Charges 5-3
Enable Payment Due with Order 5-4
Set Profile Options 5-7
Set Up Additional Line Types 5-8
Create SFM-Enabled Order Header and Line Workflows 5-8
Remove Fulfill Node in Line Workflow 5-9
Design Tips 5-10

6 Set Up Oracle Inventory and the Item Master

Overview of Set Up Oracle Inventory and the Item Master 6-1
Set Up Items 6-2
Set Up Periodicity for Recurring Charges 6-3
Create Container Model 6-4
Set Items as Provisionable 6-6
Set Up Link Items 6-6

7 Set Up Oracle Bills of Material

Overview of Set Up Oracle Bills of Material Chapter 7-1
Set Up Oracle BOM 7-1

8 Set Up Oracle Installed Base

Overview of Set Up Oracle Installed Base Chapter 8-1
Enable Network Configurations 8-2
Create Extended Attributes 8-2
Map Extended Attributes to Items 8-4
Set Up SFM Event Manager Queue Service 8-4
Design Tips 8-5

9 Set Up Oracle Advanced Pricing

Overview of Set Up Oracle Advanced Pricing Chapter 9-1
Set Up Recurring and One-Time Charges 9-2
Advisory: Order Amount and Cross-Order Volume Computations 9-3
Set Up Pricing Attributes and Sourcing Rules 9-3

10 Set Up Oracle Configurator and Customize the Solution

Overview of Set Up Oracle Configurator Chapter 10-1
Configurator Concepts 10-2
Configurator Implementation in the Solution 10-19
Disable Pricing 10-21
Map Install Base Extended Attributes to Configuration Attributes 10-22
Oracle Configurator Schema Customizations 10-24
Initialization Parameters 10-28
Batch Validation Parameters 10-29
Programmatic Tools for TSO Instance Management 10-30

11 Set Up Oracle Configurator Extensions

Overview of Set Up Oracle Configurator Extensions Chapter 11-1
General Setup for Configurator Extensions 11-3
Interactive Location Search Configurator Extension 11-6
Line Type Configurator Extension 11-13
IBAttribute Configurator Extension 11-26
Using the Oracle Configuration Interface Object (CIO) 11-29

12 Set Up Oracle Contact Center

Overview of Set Up Oracle Contact Center Chapter 12-1
Actions Architecture Overview 12-2
Set Up Additional Actions 12-2

13 Set Up Oracle iStore

Overview of Set Up Oracle iStore Chapter 13-1
Set Up Payment Due with Order 13-2
Set Profile Options 13-3
Set Up Items 13-5
Perform Pricing and Payment Setups 13-7
Perform User Setups 13-7

14 Set Up Oracle Quoting

Overview of Set Up Oracle Quoting Chapter 14-1
Set Profile Options 14-2
Set Up Recurring Charges 14-4
Set Up Payment Due with Order 14-5
Customize Quoting UI for TSO 14-5
Design Tips 14-6

15 Set Up Oracle Service Fulfillment Manager

Overview of Set Up Oracle Service Fulfillment Manager Chapter 15-1
Create SFM-Enabled Workflow Header Process 15-2
Create SFM-Enabled Workflow Line Process 15-3
Add Function for Menu Notifications 15-3
Create PL/SQL Procedure to Update Active Flag 15-4
Create Workflow Process 15-5
Define Work Item 15-6
Map Line Types 15-7
Enable Order Cancellation 15-8
Design Tips 15-10

16 Using the Oracle TSO Solution

Overview of Using the Oracle TSO Solution Chapter 16-1
Using Oracle Contact Center 16-1
Using Oracle iStore 16-3
Using Oracle HTML Quoting 16-13
Using Oracle Forms Quoting 16-18
Using Oracle Order Management 16-23

17 Administering the Oracle TSO Solution

Overview of Administering the Oracle TSO Solution Chapter 17-1
Re-Validate Installed Configurations After Modifications 17-1
Managing Notification Errors 17-3
Find an Order in SFM Order Flow-Through Area 17-3
Retry Install Base Updates 17-4
Retry Failed Outgoing Messages 17-4
Manage Failure Notifications in HTML 17-5

A Assumptions and Restrictions

Overview of Assumptions and Restrictions Appendix A-1
General Assumptions A-1
TSO With Equipment Assumptions A-3
Payment Due With Order Assumptions A-4
Recurring Charges Assumptions A-4
Oracle Configurator Integration Assumptions A-4
Oracle Inventory Integration Assumptions A-5
Oracle Order Management Integration Assumptions A-5
Oracle Quoting Integration Assumptions A-6
Oracle Service Fulfillment Manager Integration Assumptions A-7

Glossary

Index

Send Us Your Comments

Oracle Telecommunications Service Ordering Process Guide, Release 12.1

Part No. E13460-02

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document. Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?
- Did you understand the context of the procedures?
- Did you find any errors in the information?
- Does the structure of the information help you with your tasks?
- Do you need different information or graphics? If so, where, and in what format?
- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).

Note: Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the new Applications Release Online Documentation CD available on My Oracle Support and www.oracle.com. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: appsdoc_us@oracle.com

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our Web site at www.oracle.com.

Preface

Intended Audience

Welcome to Release 12.1 of the *Oracle Telecommunications Service Ordering Process Guide.*

This guide is intended for those responsible for implementing and administering the Oracle Telecommunications Service Ordering (TSO) solution.

This guide assumes you have a working knowledge of the following:

- The principles and customary practices of your business area.
- Oracle Advanced Pricing
- Oracle Bills of Material
- Oracle Configurator
- Oracle Installed Base
- Oracle Inventory
- Oracle Order Management
- Oracle Self-Service Web Applications
    - To learn more about Oracle Self-Service Web Applications, read the *Oracle Self-Service Web Applications Implementation Manual*.
- Oracle Applications Framework
- The Oracle Applications graphical user interface
    - To learn more about the Oracle Applications graphical user interface, read the *Oracle Applications User's Guide*.

If you have never used these Oracle Applications, Oracle suggests you attend one or more training classes available through Oracle University.

See Related Information Sources on page xiii for more Oracle Applications product information.

TTY Relay Access to Oracle Support Services

To reach AT&T Customer Assistants, dial 711 or 1.800.855.2880. An AT&T Customer Assistant will relay information between the customer and Oracle Support Services at 1.800.223.1711. Complete instructions for using the AT&T relay services are available at http://www.consumer.att.com/relay/tty/standard2.html. After the AT&T Customer Assistant contacts Oracle Support Services, an Oracle Support Services engineer will handle technical issues and provide customer support according to the Oracle service request process.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at http://www.oracle.com/accessibility/.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Structure

1 Introduction to Oracle TSO
2 Oracle TSO Business Processes
3 Oracle TSO Functionality

4 Implementing the Oracle TSO Solution
5 Set Up Oracle Order Management
6 Set Up Oracle Inventory and the Item Master
7 Set Up Oracle Bills of Material
8 Set Up Oracle Installed Base
9 Set Up Oracle Advanced Pricing
10 Set Up Oracle Configurator and Customize the Solution
11 Set Up Oracle Configurator Extensions
12 Set Up Oracle Contact Center
13 Set Up Oracle iStore
14 Set Up Oracle Quoting
15 Set Up Oracle Service Fulfillment Manager
16 Using the Oracle TSO Solution
17 Administering the Oracle TSO Solution
A Assumptions and Restrictions
Glossary

Related Information Sources

You can choose from many sources of information, including online documentation, training, and support services, to increase your knowledge and understanding of Oracle Telecommunications Service Ordering.

Integration Repository

The Oracle Integration Repository is a compilation of information about the service endpoints exposed by the Oracle E-Business Suite of applications. It provides a complete catalog of Oracle E-Business Suite's business service interfaces. The tool lets users easily discover and deploy the appropriate business service interface for integration with any system, application, or business partner.

The Oracle Integration Repository is shipped as part of the E-Business Suite. As your instance is patched, the repository is automatically updated with content appropriate for the precise revisions of interfaces in your environment.

Online Documentation

All Oracle Applications documentation is available online (HTML or PDF).

- **PDF** - PDF documentation is available for download from the Oracle Technology Network at http://otn.oracle.com/documentation.
- **Online Help** - Online help patches (HTML) are available on My Oracle Support.
- **About Documents and Release Notes** - Refer to the About Document or Release Notes for the mini-pack or family pack that you have installed to learn about new documentation or documentation patches that you can download. About Documents and Release Notes are available on My Oracle Support.

- **My Oracle Support Knowledge tab** - The My Oracle Support Knowledge tab lets you browse the knowledge base by technology, industry, integration, application, documentation, training, and services, to find all documents for a product area. Use the Knowledge Home to search for release-specific information, such as FAQs, recent patches, alerts, white papers, troubleshooting tips, and other archived documents.

- **Oracle E-Business Suite Electronic Technical Reference Manuals -** Each Electronic Technical Reference Manual (eTRM) contains database diagrams and a detailed description of database tables, forms, reports, and programs for a specific Oracle Applications product. This information helps you convert data from your existing applications and integrate Oracle Applications data with non-Oracle applications, and write custom reports for Oracle Applications products. Oracle eTRM is available on My Oracle Support as part of Online Documentation.

Guides Related to All Products

Oracle Applications User's Guide

This guide explains how to enter data, query, run reports, and navigate using the graphical user interface (GUI) of an Oracle Applications product. This guide also includes information on setting user profiles, as well as running and reviewing reports and concurrent programs.

You can access this guide online by choosing "Getting Started with Oracle Applications" from any Oracle Applications help file.

Guides Related to This Product

Oracle Advanced Pricing Implementation Manual

This guide show you how to define pricing rules that service the pricing requirements of Oracle applications. Pricing rules control the pricing actions that are applied to customer transactions such as price lists, price agreements, formulas, and modifiers. With Oracle Advanced Pricing, you can attach qualifiers to a price list that enables an item have more than one price list. You can also set up block pricing, promotional limits, multiple currency conversion, and multiple contexts per sales order.

Oracle Advanced Pricing User's Guide

Oracle Advanced Pricing calculates prices including promotional prices for Oracle Order Management and other Oracle Applications based on pricing rules, pricing relationships, item hierarchies, usage brackets, and deals and promotions.

Oracle Bills of Material User's Guide

Oracle Manufacturing and Oracle Order Management use bills of material to store lists of items that are associated with a parent item and information about how each item is related to its parent. Oracle Manufacturing supports standard, model, option class, and planning bills of material.

Oracle Installed Base User Guide

Oracle Installed Base is an item instance life cycle tracking application that facilitates enterprise-wide life cycle item management and tracking capability. Several Oracle Order Management transactions interact with Oracle Inventory and Oracle Installed Base.

Oracle Inventory User's Guide

This guide enables you to configure the Oracle Inventory structure to best represent your company's inventory sites and business units after you have defined your required ledger and key flexfields. You can also learn about centralized and decentralized inventory structures, and controls and reference options for using and maintaining inventory items such as categories, commodity codes, attributes, statuses, relationships, and picking rules.

Oracle iStore Implementation and Administration Guide

This guide enables your business to establish business-to-business (B2B) and business-to-consumer (B2C) electronic commerce (e-commerce). You can learn how to build, test, and launch sophisticated online stores in multiple languages and currencies that can store customer data, provide flexible pricing, track shopping lists, orders and returns, and target different customer segments and organizations. Use this guide to integrate Oracle iStore with other Oracle applications such as Oracle Workflow or Oracle Quoting for added functionality and features.

Oracle Order Management User's Guide

This guide provides information on how to use Oracle Order Management. Use this guide to learn how to enter, view and update Sales Orders, maintain sales agreements, and define pricing. This guide also explains the Order Management process and the user procedures related to each process.

Oracle Receivables User Guide

This guide provides you with information on how to use Oracle Receivables. Use this guide to learn how to create and maintain transactions and bills receivable, enter and apply receipts, enter customer information, and manage revenue. This guide also includes information about accounting in Receivables. Use the Standard Navigation Paths appendix to find out how to access each Receivables window.

Oracle Quoting Implementation Guide

Oracle Quoting enables organizations propose product solutions and negotiate prices, while enforcing consistent business rules throughout the sales cycle. This guide describes how to set up quote statuses to indicate the evolution of a quote to an order, define status transition rules, set up defaults for an operating unit, and define credit check rules. You must also set up integration with Oracle Sales Contracts, Oracle Incentive Compensation, Oracle Territory Manager, and Oracle Approvals Management.

Oracle Quoting User Guide

Oracle Quoting enables you to create and manage customer quotes across all sales and interaction channels and if approved by the customer, convert these quotes into orders. This guide describes how to create and modify quote templates, add and configure simple and related products, manage pricing and pricing adjustments, determine customer credit worthiness, and perform real-time global product availability checks.

Oracle Service Fulfillment Manager Implementation Guide

This guide describes how to define the services and service packages that your organization needs, associate actions with services, associate work items with service actions, associate fulfillment actions with work items, and associate fulfillment procedures and network elements with fulfillment actions. It also describes the business rules that you must define for PL/SQL procedures that select work items to execute for a given service, fulfillment actions to execute for a given work item, and workflow processes to execute for a given work item. You must also implement the connection logic in the connect/disconnect procedure that Oracle Service Fulfillment Manager executes to establish a communication channel with the fulfillment element via adapters.

Oracle Telecommunications Billing Integrator Implementation Guide

Oracle Telecommunications Billing Integrator provides a seamless integration between Oracle E-Business Suite and one or more external billing applications. This enables customer care agents process telecommunication service requests and billing inquiries by providing access to bill summary and customer bill images from the billing application. Oracle Telecommunications Billing Integrator publishes details on accounts, products, and sales orders to the external billing applications and integrates with the Oracle Collaboration History to monitor communication between Oracle E-Business Suite and the billing applications. This guide describes how to set up XML Gateway and the Oracle Applications InterConnect hub that contains all the mapping and transformation data required for integration.

Oracle TeleService Implementation and User Guide

This guide provides information on how to use Oracle TeleService. Oracle TeleService

automates the call center and the resolution process from the time a customer calls in, sends an email, or enters a service request on the Web. Agents can use Oracle TeleService to update customer records, validate product ownership and contract coverage, provide proactive and personalized customer service, and resolve problems that arise from the initial contact using a knowledge base.

Installation and System Administration

Oracle Applications Concepts

This book is intended for all those planning to deploy Oracle E-Business Suite Release 12, or contemplating significant changes to a configuration. After describing the Oracle Applications architecture and technology stack, it focuses on strategic topics, giving a broad outline of the actions needed to achieve a particular goal, plus the installation and configuration choices that may be available.

Oracle Applications Installation Guide: Using Rapid Install

This book is intended for use by anyone who is responsible for installing or upgrading Oracle Applications. It provides instructions for running Rapid Install either to carry out a fresh installation of Oracle Applications Release 12, or as part of an upgrade from Release 11*i* to Release 12. The book also describes the steps needed to install the technology stack components only, for the special situations where this is applicable.

Oracle Applications Maintenance Utilities

Use this guide to help you run the various Applications DBA (AD) utilities, such as AutoUpgrade, AutoPatch, AD Administration, AD Controller, AD Relink, License Manager, and others. It contains how-to steps, screenshots, and other information that you need to run the AD utilities. This guide also provides information on maintaining the Oracle Applications file system and database.

Oracle Alert User's Guide

This guide explains how to define periodic and event alerts to monitor the status of your Oracle Applications data.

Oracle Applications Developer's Guide

This guide contains the coding standards followed by the Oracle Applications development staff. It describes the Oracle Application Object Library components needed to implement the Oracle Applications user interface described in the *Oracle Applications User Interface Standards for Forms-Based Products*. It also provides information to help you build your custom Oracle Forms Developer forms so that they integrate with Oracle Applications. In addition, this guide has information for customizations in features such as concurrent programs, flexfields, messages, and logging.

Oracle Applications User Interface Standards for Forms-Based Products

This guide contains the user interface (UI) standards followed by the Oracle Applications development staff. It describes the UI for the Oracle Applications products and tells you how to apply this UI to the design of an application built by using Oracle Forms.

Other Implementation Documentation

Oracle Workflow Administrator's Guide

This guide explains how to complete the setup steps necessary for any Oracle Applications product that includes workflow-enabled processes, as well as how to monitor the progress of runtime workflow processes.

Oracle Workflow Developer's Guide

This guide explains how to define new workflow business processes and customize existing Oracle Applications-embedded workflow processes. It also describes how to define and customize business events and event subscriptions.

Oracle Workflow User's Guide

This guide describes how Oracle Applications users can view and respond to workflow notifications and monitor the progress of their workflow processes.

Oracle Applications Flexfields Guide

This guide provides flexfields planning, setup and reference information for the Oracle Projects implementation team, as well as for users responsible for the ongoing maintenance of Oracle Applications product data. This guide also provides information on creating custom reports on flexfields data.

Oracle E-Business Suite Diagnostics User's Guide

This guide contains information on implementing, administering, and developing diagnostics tests in the Oracle E-Business Diagnostics framework.

Oracle E-Business Suite Integrated SOA Gateway Implementation Guide

This guide explains the details of how integration repository administrators can manage and administer the entire service enablement process based on the service-oriented architecture (SOA) for both native packaged public integration interfaces and composite services - BPEL type. It also describes how to invoke Web services from Oracle E-Business Suite by working with Oracle Workflow Business Event System, manage Web service security, and monitor SOAP messages.

Oracle E-Business Suite Integrated SOA Gateway User's Guide

This guide describes how users can browse and view the integration interface definitions and services that reside in Oracle Integration Repository.

Training and Support

Training

Oracle offers a complete set of training courses to help you master your product and reach full productivity quickly. These courses are organized into functional learning paths, so you take only those courses appropriate to your job or area of responsibility.

You have a choice of educational environments. You can attend courses offered by Oracle University at any of our many Education Centers, you can arrange for our trainers to teach at your facility, or you can use Oracle Learning Network (OLN), Oracle University's online education utility. In addition, Oracle training professionals can tailor standard courses or develop custom courses to meet your needs. For example, you may want to use your organization structure, terminology, and data as examples in a customized training session delivered at your own facility.

Support

From on-site support to central support, our team of experienced professionals provides the help and information you need to keep your product working for you. This team includes your Technical Representative, Account Manager, and Oracle's large staff of consultants and support specialists with expertise in your business area, managing an Oracle server, and your hardware and software environment.

Do Not Use Database Tools to Modify Oracle Applications Data

Oracle STRONGLY RECOMMENDS that you never use SQL*Plus, Oracle Data Browser, database triggers, or any other tool to modify Oracle Applications data unless otherwise instructed.

Oracle provides powerful tools you can use to create, store, change, retrieve, and maintain information in an Oracle database. But if you use Oracle tools such as SQL*Plus to modify Oracle Applications data, you risk destroying the integrity of your data and you lose the ability to audit changes to your data.

Because Oracle Applications tables are interrelated, any change you make using an Oracle Applications form can update many tables at once. But when you modify Oracle Applications data using anything other than Oracle Applications, you may change a row in one table without making corresponding changes in related tables. If your tables get out of synchronization with each other, you risk retrieving erroneous information and you risk unpredictable results throughout Oracle Applications.

When you use Oracle Applications to modify your data, Oracle Applications

automatically checks that your changes are valid. Oracle Applications also keeps track of who changes information. If you enter information into database tables using database tools, you may store invalid information. You also lose the ability to track who has changed your information because SQL*Plus and other database tools do not keep a record of changes.

1

Introduction to Oracle TSO

This chapter covers the following topics:

- Overview of Oracle Telecommunications Service Ordering

Overview of Oracle Telecommunications Service Ordering

Ordering and delivering telecommunication services can be quite complex, and this complexity only increases with service changes. The telecommunications service ordering (TSO) functionality in Oracle Applications supports the processes involved in ordering and reconfiguring or changing (including moving, adding, changing, and disconnecting [MACD] customer services).

Oracle Applications reduce the complexity of the ordering process by providing:

- A centralized view of the customer, including the installed service configuration
- A repository of the products and services available to the customer
- The ability to offer customer quotes and flexible pricing
- The ability to restore, reconfigure, and re-price an existing service configuration
- Full order management and processing capabilities
- Online ordering through Oracle iStore

The Oracle Telecommunications Service Ordering (TSO) solution enables you (or customers themselves, in self-service online mode) to create new or update existing configurations of telecommunication equipment and services by moving, adding, changing, or disconnecting a customer's services and equipment. To enable the upgrade process flow, companies need the ability to track the full life cycle of an item instance -- product information, location, status, and so on -- from order creation to order fulfillment across multiple orders. If a customer requests an upgrade of a product from a company after several years have passed, the company must know the exact current

state of that product and must verify that any requested changes are feasible, such as whether or not these changes are compatible with the customer's existing installed product, and fulfilling only those requested changes.

Applications Involved in the Oracle TSO Solution

The TSO solution spans multiple Oracle applications, including:

- Oracle Accounts Receivable
- Oracle Advanced Pricing
- Oracle Advanced Product Catalog
- Oracle Configurator
- Oracle Contact Center
- Oracle Bills of Material
- Oracle iStore
- Oracle Installed Base
- Oracle Inventory
- Oracle Order Management
- Oracle Quoting
- Oracle Service Fulfillment Manager
- Oracle Telecommunications Billing Integrator

Depending upon your business requirements, other Oracle applications may also be required. For a list of mandatory applications, refer to the section, "Mandatory Dependencies for the Oracle TSO Solution", in the chapter, Implementing the Oracle TSO Solution, page 4-1.

Each application has developed a set of functionality, which, when used together with the other applications, enables the Oracle TSO solution. See the chapter, Oracle TSO Business Processes, page 2-1, for a summary of TSO functionality by Oracle application.

2

Oracle TSO Business Processes

This chapter covers the following topics:

- Oracle TSO Business Processes Overview
- Configuring New Item Instances
- Reconfiguring Installed Item Instances
- Disconnecting Installed Item Instances
- Changing Item Instances from Oracle Installed Base
- Pricing a Configuration
- Placing the Order and Verifying Order Fulfillment
- Integration with Oracle Telecommunications Billing Integrator

Oracle TSO Business Processes Overview

The Oracle Telecommunications Service Ordering (TSO) solution consists of the following processes:

- Create new configuration of equipment and/or services
- Reconfigure installed equipment and/or services
- Disconnect installed equipment and/or services

You can initiate these processes from Oracle iStore, Oracle Quoting, Oracle Contact Center (TeleService), or Oracle Order Management.

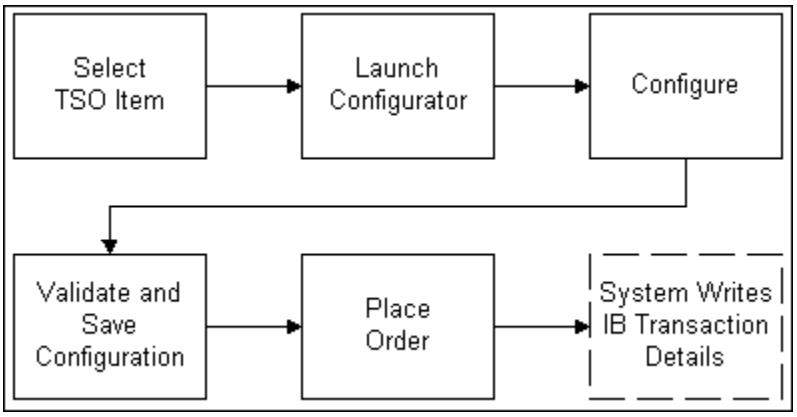
Configuring New Item Instances

The process of configuring a new item begins when a user selects a configurable item from the Oracle Inventory catalog and launches Oracle Configurator. In Configurator, the user selects options available in the Container Model and then creates a quote or

order line.

The following figure shows the process of configuring new items.

Process for Configuring New Item Instances



The following paragraphs explain the above figure.

Process for Configuring New Item Instances

1. From the product catalog or Oracle Inventory, the user selects the TSO item. Each selected item instance creates a quote line or order line. This may occur, for example, in Oracle Quoting, where the user is a sales agent creating a quote (and eventually an order) for a customer.

2. In the configuration process, the hosting application launches Oracle Configurator, and the user configures the item. For example, a customer in Oracle iStore may select a telephone to configure; in the Configurator, the user will add options to the phone, such as a service plan or a warranty. When finished configuring the item, the user places the order with the item instance. For example, an agent in Oracle Contact Center may place the order for a customer.

   During the Oracle Configurator session, users can configure pick-to-order (PTO) model items by selecting their attributes, locations, connections between items, and by specifying the instance names for the selected items. Oracle Configurator identifies -- for every item in the configuration -- a provisioning action (for example, add or disconnect). Oracle Configurator validates the configuration against defined technical and business configuration rules.

3. When the user is finished with the configuration, he submits the configuration, which results in:

   - The Oracle Configurator window closing and:
     - Passing the changes back to the hosting application
     - Passing information about the transaction lines associated with each item to

Oracle Installed Base

- Multiple lines, selected during the configuration session, appearing in the quote or order

4. The item instance is then provisioned. In other words, an automatic or manual process writes all of the details of the item instance. For example:
   - The phone number and customer information are recorded into systems and databases
   - A DSL technician comes to the customer's house and installs software and a modem
5. Oracle Configurator writes transaction details of the configuration to Oracle Installed Base.
6. After provisioning, install base writing, and fulfillment, the item instance can be reconfigured.

When Oracle Service Fulfillment Manager has fulfilled an order line, Oracle Installed Base is updated with any changes to extended attributes that occurred during fulfillment.

Integration with Oracle Financials applications provides billing/invoicing supporting, and integration with Oracle Telecommunications Billing Integrator provide links to third-party billing systems and two-way communication with Oracle E-Business Suite objects such as product retrieval and orders.

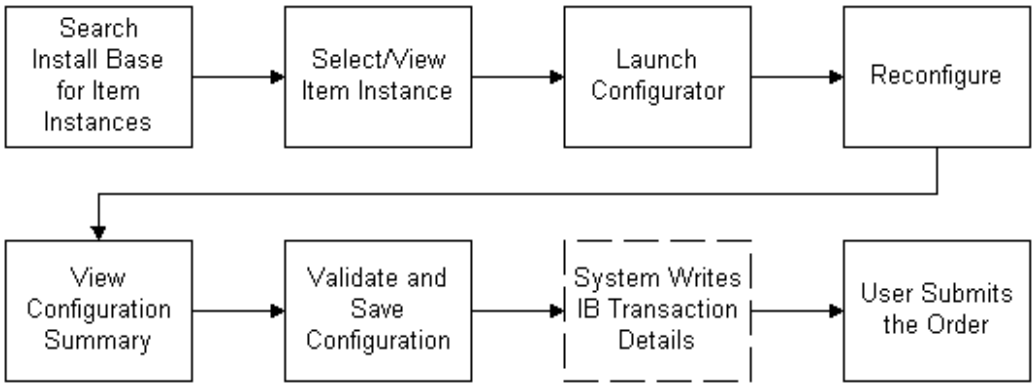
Reconfiguring Installed Item Instances

From within a hosting application (e.g., Oracle Quoting) the user begins the process of reconfiguring an item instance by selecting a configurable item instance from the customer's installed base. Item instances created in a common Container Model appear as separate quote/order lines, as child lines of a single parent container line. Each instance shows up on a separate line, but each line is a child of a single parent line.

If integrating with Oracle iStore, customers reconfigure their own install base items in the Oracle iStore Customer Application. This flow is covered in the chapter, Oracle TSO Functionality, page 3-1.

The following figure shows the process of reconfiguring item instances.

Process for Reconfiguring Installed Item Instances



The following paragraphs explain the above figure.

Prerequisites: An order has been placed for an item instance, the item instance has been provisioned, and the details have been written to the customer's install base.

Reconfiguring Installed Item Instances

1. After an order has been placed for a item instance and the details written to the customer's installed base, ordered item can be reconfigured using one of the hosting applications. In this process, the user begins by searching Oracle Installed Base from within the hosting application. The user selects the item. For example, from within Oracle Quoting, a sales agent can search a customer's installed base items and select an item to reconfigure.

2. The user selects the item instance, or a subset of the product instances, to move, add or change -- that is, reconfigure. The selected item instances appear in the quote or order line within their original Container Model.

3. The user launches Oracle Configurator. Oracle Configurator retrieves the *baseline configuration* status from Oracle Installed Base. The baseline configuration represents the latest revision of a configurable item and reflects the current installation at a customer's site.

4. The user reconfigures item instances. For example, he changes attributes, locations, and connections between item instances, or he moves or disconnects the item instance or adds new item instances. Oracle Configurator validates the changed configuration against defined configuration rules.

5. Optionally, in sales agent-enabled applications, the user can compare the reconfiguration to the previous configuration in the Configurator summary window. The summary window shows the changes since the previously fulfilled, baseline configuration.

   The summary window indicates the type of change in the Line Type column. When an Oracle Quoting agent submits a quote, creating a sales order in Oracle Order

Management, the provisioning action for each quote and order line appears in the Line Type column in the Oracle Order Management Sales Orders window. Downstream applications, such as Oracle Service Fulfillment Manager, also reference the line type for further processing.

6. After the user is satisfied with and has completed the configuration, the Oracle Configurator session ends, and the quote or order line displays the details of the configuration.

7. The user places the order. The quote/order contains the delta between the baseline configuration and the revised one.

8. The reconfigured item instance is then provisioned. In other words, an automatic or manual process writes all of the details of the item instance. For example:
   - The phone number and customer information are recorded into systems and databases
   - A DSL technician comes to the customer's house and installs software and a modem

9. Oracle Configurator writes transaction details of the configuration to the Oracle Installed Base.

10. After provisioning, install base writing, and fulfillment, the item instance can be reconfigured again.

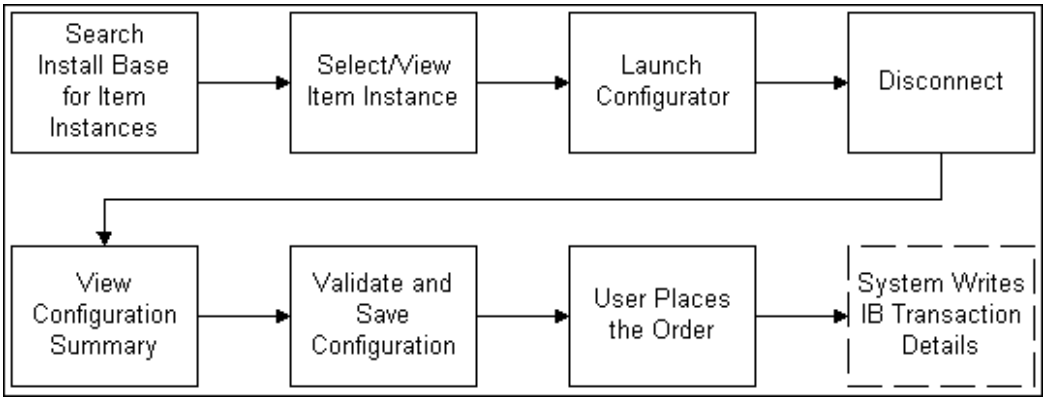
Note: In the telecommunications service industry, it is common to use the term, *upgrade*, to refer to the reconfiguration of a service by moving, adding, changing, or disconnecting items (MACD). However, in the context of the TSO solution, these changes to installed configurations of Container Models are regarded as *updates* to those configurations. To avoid confusion, this document uses the term, *reconfiguration*, to encompass both *upgrade* and *update*.

Disconnecting Installed Item Instances

The Oracle TSO solution supports the ability of customers to disconnect installed items instances.

The following figure shows the process of disconnecting item instances.

Process for Disconnecting Installed Item Instances



The following paragraphs explain the above figure.

Prerequisites: An order has been placed for an item instance, the item instance has been provisioned, and the details have been written to the customer's install base.

1. After an order has been placed for a item instance and the details written to the customer's installed base, ordered item can be disconnected using one of the hosting applications. In this process, the user begins by searching Oracle Installed Base from within the hosting application. The user selects the item. For example, from within Oracle Order Management, a sales agent can search a customer's installed base items and select an item to disconnect.

2. The user selects the item instance, or a subset of the product instances, to disconnect. The selected item instances appear in the quote or order line within their original Container Model.

3. The user launches Oracle Configurator. Oracle Configurator retrieves the *baseline configuration* status from Oracle Installed Base. The baseline configuration represents the latest revision of a configurable item and reflects the current installation at a customer's site.

4. The user disconnects the item instance.

5. Optionally, in sales agent-enabled applications, the user can compare the reconfiguration to the previous configuration in the Configurator summary window. The summary window shows the changes since the previously fulfilled, baseline configuration.

6. After the user is satisfied with and has completed the configuration, the Oracle Configurator session ends, and the quote or order line displays the details of the configuration.

7. The user places the order. The quote/order contains the delta between the baseline

configuration and the revised one.

8. The disconnected item instance is then provisioned. For example, the telephone company updates its records that a customer's telephone number has been removed from service.

9. Oracle Configurator writes transaction details of the configuration to the Oracle Installed Base.

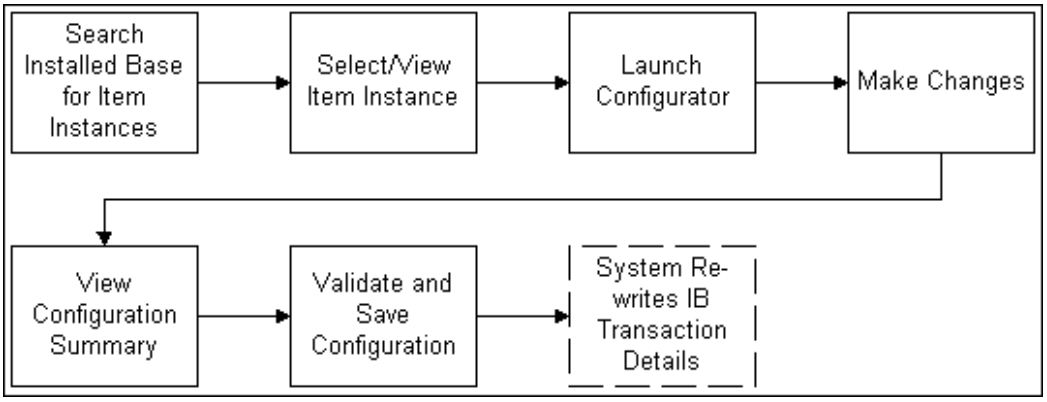
Changing Item Instances from Oracle Installed Base

In Oracle Installed Base, the user can make changes to an installed item directly, without involving one of the hosting applications.

Use case example: A customer has placed an order for DSL 512KB speed. The DSL was installed as ordered, but the customer receives a bill for a 1MB speed rate. In this case wherein the system records are incorrect, a customer service representative can go directly into the installed base and update the records manually. When users make changes directly in the installed base, provisioning does not take place.

The following figure shows the process of changing an item instance from within Oracle Installed Base.

Process for Changing Item Instances from Installed Base



The following paragraphs explain the above figure.

Prerequisites: An order has been placed for an item instance, the item instance has been provisioned, and the details have been written to the customer's install base.

1. In Oracle Installed Base, the user searches for installed item instances.

2. The user selects an item to reconfigure.

3. The user launches Oracle Configurator and reconfigures the item.

4. The user views the configuration summary.

5. The user validates and saves the reconfiguration.

6. The system writes the installed base transaction details into the customer's installed base.

Pricing a Configuration

When a configuration is completed, validated, and saved in Oracle Configurator, the configuration returns to the hosting application. Quote or order lines are created for each item instance in the configuration. The hosting application then calls Oracle Advanced Pricing to price the configuration (using the pricing setup for the hosting application, along with any setups done in Oracle Advanced Pricing). The prices of each item appear on each quote or order line.

In Oracle iStore, the prices appear in the shopping cart with the configuration. In the sales-agent enabled applications, agents can adjust prices and other quote or order information before placing an order.

For more information on placing orders in these applications, refer to the respective product documentation.

Placing the Order and Verifying Order Fulfillment

From the hosting application or Oracle Installed Base, the user places the order with the item instance.

To prevent data corruption, Oracle Configurator locks instances until they are completely fulfilled, to ensure the integrity of their revision data. For details, see the chapter, Set Up Oracle Configurator and Customize the Solution, page 10-1.

Oracle Service Fulfillment Manager (if implemented) receives the order from Oracle Order Management for provisioning. When the system provisions the order, Oracle Installed Base receives the reconfigured item information, and the sales order is complete.

When Oracle Service Fulfillment Manager completes provisioning, the order line status becomes Provisioning Successful, and then it becomes Fulfilled.

Integration with Oracle Telecommunications Billing Integrator

Implementers can use Oracle Telecommunications Billing Integrator (TBI) to integrate with the third-party billing systems. The third-party billing systems rate and generate bills for the services ordered. Oracle TBI provides two-way integration which enables exchange of information between Oracle E-Business Suite and the external billing applications. For more information, see the chapter, Oracle TSO Functionality, page 3-

3

Oracle TSO Functionality

This chapter covers the following topics:

- Overview of Key Functionality
- Oracle Configurator Functionality
- Oracle Installed Base Functionality
- Functionality of Sales Agent-Enabled Applications and Oracle iStore
- Oracle Advanced Pricing Functionality
- Oracle Service Fulfillment Manager Functionality
- Oracle Telecommunications Billing Integrator Functionality

Overview of Key Functionality

The Oracle Telecommunications Service Ordering (TSO) solution spans multiple Oracle applications. Each application provides a set of functionality that, when used together, enables the Oracle TSO solution. At a high level, the key functionality involved and supported in the Oracle TSO solution are discussed in this chapter.

Recurring and One-Time Charges

Recurring charges are fixed charges that are applied to a customer's account on a recurring basis. Examples of recurring charges are $29.99 per month for a wireless phone service or $5000 per year for a full T1 line. Recurring charges are always considered Pay Later amounts. The *period of recurrence* for the phone service is every month, and the period of recurrence for the full T1 is once a year. The recurring price of a service is captured as the unit list price of the item representing the service. The *charge periodicity* of the item representing the service is the period of recurrence for the charge. When setting up prices for recurring charge items, implementers select charge periodicity in the pricing attribute field on the price list line. The charge periodicity attribute indicates the period of recurrence (for example, if the price is per month or per year).

A one-time charge is a fixed charge that a customer pays the telecommunications service provider only once. Some examples of one-time charges include activation fees, installation fees, and fees to switch long-distance carriers.

The hosting applications support recurring charges functionality in order to price and sell products with recurring and one-time charges.

Discounts on Recurring Charges

In Oracle Quoting, Oracle Contact Center, and Oracle Order Management, sales agents can make discounts to recurring charges.

- **Oracle Quoting:** Agents can apply discounts to the recurring list price of a service by specifying the discount in the Oracle Quoting UI (Forms and HTML versions). For more information, see the *Oracle Quoting User Guide*.

- **Oracle Order Management:** Agents can use the standard Oracle Order Management line discount functionality to discount recurring charges on an order.

Setting up Recurring and One-Time Charges

Implementers should use separate items to model recurring and one-time charges. The items with recurring and one-time charges must be price list items with prices greater than zero.

The one-time charge must be flagged as a *transient item*, that is, an item that does not persist in a configuration after fulfillment. For more information, see the chapter, Set Up Configurator and Customize the Solution, page 10-1.

The subtotals are the sum of the one-time and recurring charges, both in Oracle Order Management and Oracle Quoting. Implementers use standard line-level discounts to provide discounts for one-time charges. The system allows implementers to add taxes on the one-time charges

Note that Oracle Configurator rules or Oracle Configurator Extensions can be used to select the one-time-charge items into the configuration.

For more information about recurring charges in hosting applications, see the chapter, Using the Oracle TSO Solution, page 16-1.

Payment Due with Order

When selling services and equipment to consumers, the customer may be required to pay for a portion of the total order amount when the order is placed and pay another amount later, based on the business logic specified by the service provider. This functionality is known as Payment Due with Order. The Payment Due with Order amount includes all items flagged with a Pay Now payment term, plus any shipping charges. Thus, the Payment Due with Order amount is based on the line-level payment term attached to the item. Implementers also can have an order-level payment term attached to the order, and this can be included in the Payment Due with Order amount.

The Payment Due with Order amount might include any of the following amounts in the cart, quote, or order:

- Equipment charges
- Freight and shipping charges
- Taxes
- One-time charges, such as activation fees, installation fees, or change fees

The Payment Due with Order amount might include recurring charges, such as monthly Internet service fees. Recurring charges are always considered "pay later" charges.

For details about the calculation and display of Payment Due with Order amounts in the hosting applications, see the chapter, Using the Oracle TSO Solution, page 16-1.

Discounts on One-Time Charges

In Oracle Quoting, Oracle Contact Center, and Oracle Order Management, sales agents can make discounts on one-time charges.

- **Oracle Quoting:** Implementers can give discounts on one-time charges. For additional details, refer to the *Oracle Quoting User Guide.*
- **Oracle Order Management:** Implementers can increase or decrease freight and special charges for a line using a manual price. Note that the freight charge, Overide Allowed, must be used to be able to manually increase or decrease a value on freight and special charges.

In both Oracle Order Management and Oracle Quoting, implementers can use the standard line discounting functionality to discount one-time charges

One-Time Charge Totals

One-time charge totals are discussed below.

- **Oracle Quoting:** The Lines page in Oracle Quoting:
  - Shows the sum of the one-time charges on the Charges row of the Grand Total section.
  - The user can view a break-down of all the one-time charges by selecting a hyperlink.

If large configurations or complex pricing exist, see the topics about setting up manual pricing and tax calculation request options in the *Oracle Quoting Implementation Guide*; and information about pricing, products, and quotes in the *Oracle Quoting User Guide*.

- **Oracle Order Management:** The sum of one-time charges appear in the charges total.

Oracle Configurator Functionality

Oracle Configurator offers the following functionality in support of Oracle TSO processes:

- Configure new items and create new configurations
- Restore configurations from Oracle Installed Base
- Reconfigure existing configurations
- Determine the differences between the installed and the reconfigured configuration
- Submit the reconfigured configuration to the hosting application

With Oracle Configurator, the Oracle TSO solution supports integration with Oracle Installed Base, the computation of deltas and Line Types, the support of attributes, and the partial reconfiguration of item instances modeled by Oracle Configurator Components (but only when the Components are children of a Container Model).

> **Important:** The ability to reconfigure installed configurations of container Models is designed for and intended for use by the TSO industry. Users in this area of business need to configure and reconfigure a large network of connected Components, such as long-distance telephone networks. Installed configurations such as these contain only serviceable items and certain tangible items; they do *not* contain items based on ATO BOM Models.

Oracle Installed Base Integration

Through integration with Oracle Installed Base, users can track the full life cycle of a product as it is uniquely configured, ordered, fulfilled, serviced, and reconfigured over time. Oracle Installed Base serves as the central repository for storing all product information. In performing any type of product reconfiguration, Oracle Configurator uses Oracle Installed Base data as the baseline when performing product reconfiguration.

Partial Network Reconfiguration and Validation

Partial network reconfiguration and validation initially loads only the item instances that the end user explicitly selects to reconfigure and restores them into the runtime configuration session. Although these selected instances might be the only ones

restored, Oracle Configurator validates any proposed changes against all affected item instances to ensure a valid state of the complete network. The selected instances are also called *active components*; the instances that might be indirectly affected are called *passive components*.

In the sales agent-enabled applications, agents can specify whether they wish to reconfigure existing services or purchase new ones. An Oracle Installed Base search window is provided, and agents can perform queries based on a variety of criteria. The agent can choose item instances from the search results to reconfigure. Oracle Configurator launches with only the active item instances and their related passive item instances in the reconfiguration session. In Oracle iStore, unchanged item instances are automatically filtered out of the UI.

The key benefits of partial network reconfiguration and validation functionality are:

- The end user can reconfigure sub-instances independently, while ensuring validity throughout the entire network.
- Multiple concurrent users can reconfigure different distantly-related item instances in the same network while maintaining the overall validity of the network.
- Runtime performance is based on the size of the item instances to be reconfigured. When a company is selling complex products, such as Frame Relay Services consisting of thousands of locations and thousands of connections, it is a significant performance advantage to be able to load only those item instances that the end user chooses to reconfigure.

Computation of Configuration Deltas

Computation of Configuration Deltas is the ability to identify the installed state of a configuration as the baseline, allow the user to reconfigure from this installed state, and to quote, price, and fulfill only those changes. The user can change these options and items to meet new requirements. The Configurator Summary page displays the changes that took place during the configuration session. The user has the option to view either:

- The changes with respect to what was selected from Oracle Installed Base
- The entire configuration, including the changes

Persistence of Line Types

Oracle Configurator associates changes with appropriate Line Types. For example, changing the configuration by adding an item results in a Line Type that might be named ADD-MACD (MACD stands for move, add, change, disconnect). You define the Line Type, ADD-MACD, when you implement the Oracle TSO solution. Furthermore, the TSO implementer can tie these Line Types to the appropriate fulfillment steps.

Persistence of Attribute Values

User-defined attribute values, such as custom names and locations, persist throughout the fulfillment process in one global instance (Oracle Installed Base), so that all applications can have access to the most up-to-date information. These values are stored and reconfigured throughout the order creation to order fulfillment process to ensure timely and accurate service fulfillment.

Oracle Installed Base Functionality

Oracle Installed Base provides the following functionality for the Oracle TSO solution:

- Stores network configurations
- Identifies and maintains the current baseline configuration
- Launches and validates updates to the baseline configuration
- Manages connected-to relationships

After the user submits the order and it is fulfilled, the saved configuration appears in Oracle Installed Base as a set of item instances. Users can view or reconfigure the configured instances, their attributes, and relationships from Oracle Installed Base, Oracle Quoting, Oracle Contact Center, Oracle iStore, or Oracle Order Management, which all access Oracle Installed Base.

For more information about Oracle Installed Base, see the *Oracle Installed Base Implementation Guide* and the *Oracle Installed Base User Guide*.

Network Configuration Models

Oracle Installed Base can support a network connection at multiple service locations. This includes networks implemented in rings; for example, a network where A is connected to B, B is connected to C, and C is connected to A.

Current Baseline Configuration

Oracle Installed Base stores the baseline configuration, which is represents the latest revision of a configurable item; the baseline configuration reflects the current installation at a customer's site.

Connected-to Relationships

Oracle Configurator uses connected-to relationships to make network connections. You can view the connected-to relationships of a configured instance from Oracle Installed Base. If you define items in Oracle Inventory as link items, you can use Oracle Installed

Base to show the start and end locations of the link. The locations for the link instance are the geographic addresses of the instance items. For information about defining an inventory item as a network link, see the chapter, Set Up Oracle Inventory and the Item Master, page 6-1.

When an implementer utilizes a Configurator Extension to set the location of a link item, the location is saved into Oracle Installed Base and is retained when the configured instance is later reconfigured. If implementers do not set the location (that is, with a Configurator Extension), then Oracle Installed Base sets the Location and Location Type Code during fulfillment, using the values from one of the targets of the link item.

Support for Item Instance Search

The Item Instance Query window displays existing configured item instances, including attributes and components, primarily to support the Oracle Order Management TSO functionality when using the Sales Order or Quick Sales Order windows. The Item Instance Query window provides integration and flexibility in entering TSO changes from multiple sources.

The key features of the window are:

- Ability to query instances based on a wide range of query parameters, including extended attributes
- Ability to view configurations, in a tree structure and select the instances to change
- When called from the Sales Order window, the ability to return a table of records for processing by Oracle Order Management

Functionality of Sales Agent-Enabled Applications and Oracle iStore

Note: When this guide refers to *sales agent-enabled applications*, it is referring to Oracle Order Management, Oracle Contact Center, and Oracle Quoting. The sales-agent-enabled applications do not include Oracle iStore, whose Customer Application is a self-service online ordering application for end customers. Collectively, these applications are known in this guide as the *hosting applications*, because they *host* Oracle Configurator Container Models.

The sales agent-enabled applications and Oracle iStore offer support for the following Oracle TSO functionality:

- **Search Oracle Inventory:** This functionality allows the searching, retrieval and configuration of products from Oracle Inventory.
- **Search Oracle Installed Base:** This functionality allows the searching, retrieval, and reconfiguration of existing installed instances from Oracle Installed Base.

- **Integration with Oracle Configurator:** This functionality allows the configuration of new instances or the reconfiguration of installed instances to the order.

- **Payment Due with Order Amount:** This functionality allows the display of the amount due with order, along with payments due at a later date. It also includes support in the back-end processing applications for this functionality.

- **Recurring Charges:** This functionality allows the display of charges that recur periodically (e.g., monthly). It also includes support in the back-end processing applications for this functionality.

- **Ordering of Equipment Items:** This functionality allows tangible items (e.g., telephones) to be ordering within a Container Model.

Configuring and Reconfiguring Installed Items

In sales agent-enabled applications, Oracle TSO functionality allows agents to add or reconfigure TSO items from a customer's install base. When a customer wishes to make changes or add components to a product configuration that he has ordered and received, agents can either search Oracle Installed Base and select the item or search Oracle Inventory to add new items. The agent can launch Oracle Configurator and either revise the original configuration or configure the new items. In the Oracle iStore Customer Application, customers can configure new TSO items or search their install base and reconfigure existing TSO items.

In Oracle Contact Center, if the sales agent disconnects an item instance, or modifies the instance name, location or extended attributes of a configured item, Oracle Configurator validates the configuration through batch validation when changes to the item are saved. It is not necessary to launch Oracle Configurator in its own UI to perform these actions. When batch validation is launched in this way by Oracle Contact Center, it does not create a permanent configuration record.

Oracle Configurator returns a Line Type (or action) to the hosting applications to inform the user when a change has been made. The Container Model and all Components have the default line category (or order). After configuration, the pricing of the quote or order occurs, after which placement of the order occurs. When the order is booked, fulfillment of order lines occurs in Oracle Service Fulfillment Manager.

When Oracle Service Fulfillment Manager has fulfilled an order line, Oracle Installed Base is updated with any changes to extended attributes that occurred during fulfillment. Integration with Oracle Financials applications provides billing/invoicing supporting, and integration with Oracle Telecommunications Billing Integrator provide links to third-party billing systems and two-way communication with Oracle E-Business Suite objects such as product retrieval and orders.

Tangible Items

In the Oracle TSO solution, a *tangible item* is a physical good. Typically, when a tangible

item is ordered through one of the hosting applications (either online or through a telephone call with a sales agent), the item is shipped to the customer. After a customer has placed an order for a TSO item, he may wish to later make changes to it (reconfigure it). To support reconfiguration of tangible items, the Oracle TSO solution leverages the Container Model (requiring integration with Oracle Configurator, Oracle Bills of Material, and Oracle Installed Base). A TSO Container Model, which contains both products and services, is a specific type of pick-to-order (PTO) model that allows services within the Model. After the order is placed, the Model resides in the customer's install base. If the tangible item is part of Container Model, only the shipped order line (tangible item) is marked as shipped. The other order lines (*intangible items*) of the same Model are untouched. If a kit that is part of a Container Model resides on an order line and the kit has been reconfigured, order lines are not created for the components of the kit. The only allowable reconfiguration line type for a kit is Disconnect. A Container Model can contain tangible standard items or kits.

For more information about tangible items in the specific hosting application, see the chapter, Using the Oracle TSO Solution, page 16-1.

Oracle Installed Base Integration

Oracle Installed Base stores the current configuration of a customer's existing products. When a customer requests a change to an existing configuration, the sales agent can search Oracle Installed Base from within the sales agent-enabled applications. After the agent selects an Oracle Installed Base instance, the system returns the instance to the quote or order. In Oracle iStore, customers can access the My Products screen within Order Tracker in the Customer Application to search their install base and reconfigure existing items.

After completing the reconfiguration, the sales agent or customer returns to the hosting application, where the Action column shows the provisioning action performed on each telecommunication service line during reconfiguration. The hosting application calls Oracle Advanced Pricing to get updated pricing for the Model and Components. In the sales agent-enabled applications, agents can remove unchanged Components from the quote or order if the customer only wishes to view changed items. For more information on unchanged Components, see the chapter, Using the Oracle TSO Solution, page 16-1. Sales agents can also control unchanged components by setting the profile option, CZ: Only Create CZ Config Items for Selected Nodes. For more information, see the chapter, Set Up Configurator and Customize the Solution, page 10-1, and the *Oracle Configurator Installation Guide*.

Oracle Quoting TSO Flow

Following is a typical TSO functionality scenario from the perspective of Oracle Quoting. This is a basic flow that covers both HTML and Forms versions of Oracle Quoting.

1. A customer contacts a sales agent (sales representative) for a new

telecommunications product or a change to an existing installed item instance.

2. The sales agent creates a quote, selecting the customer in the Oracle Quoting Search and Select: Customer page.

3. If configuring a new product, the sales agent searches for and selects a product from available products in Inventory. If reconfiguring an existing item instance, the agent searches the customer's installed base and selects an installed item instance. The agent adds the item instance to the quote.

4. The sales agent launches Oracle Configurator, allowing initial configuration or reconfiguration of the item instance. Available actions and modifications (move, add, change, disconnect) are controlled by Oracle Configurator rules.

5. After the configuration or reconfiguration is complete, the sales agent returns to Oracle Quoting, where the Action line type shows the action performed on each telecommunication service line during reconfiguration.

6. Oracle Quoting calls Oracle Pricing to get updated pricing for the item instance. In the case of reconfiguration, the sales agent can remove unchanged components; this would be helpful, for example, if the customer wishes to know the change in price due solely to reconfiguration.

7. The sales agent submits the order to Oracle Order Management.

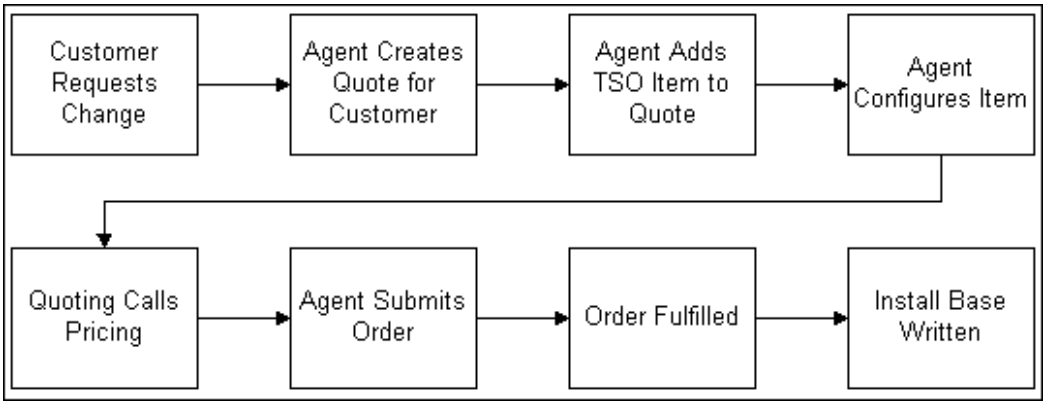
8. The item instance is provisioned.

9. Oracle Service Fulfillment retrieves the order from Oracle Order Management and fulfills it.

10. Once order fulfillment is complete, the customer's installed base is updated with the completed changes.

The following figure shows the flow.

Oracle Quoting TSO Flow



Oracle iStore TSO Flows

Oracle iStore supports telecommunications service ordering (TSO) in the Customer Application. The Customer Application is a complete ordering system containing the online store(s) implemented by an organization. Typical TSO functionality in the Customer Application involves selling telecommunications products (e.g., cell phones) and their related service plans to customers online. To present the items, Oracle iStore leverages its integration with Oracle Bills of Material to build the models and integration with Oracle Configurator to offer customers the ability to configure and re-configure the items. In the Site Administration Application, merchants add the TSO items and related service plans to the catalog and offer them for sale in the stores.

Support for the Oracle TSO solution in Oracle iStore includes allowing customers to renew calling plans online, reconfigure existing items from the install base, and track items ordered. On the merchant side, support is provided for Payment Due with Order functionality and the appropriate representation of configured items.

Following are the high-level process flows for TSO functionality from the Oracle iStore perspective:

Oracle iStore Flow for New Configuration

Following is the high-level process flow for a customer to configure a TSO item for the first time:

1. A customer browses the catalog in an Oracle iStore Customer Application online store and selects a product (a physical good) and an accompanying service package from the catalog.

2. The customer presses the Configure button next to the item description.

3. Oracle iStore launches Oracle Configurator, and the customer configures the item, selecting the optional features of the service package.

4. When the customer is finished with the configuration, the configuration is validated in Oracle Configurator and considered complete.

5. The system returns the customer to the shopping cart, where the configured items are displayed in a hierarchy.

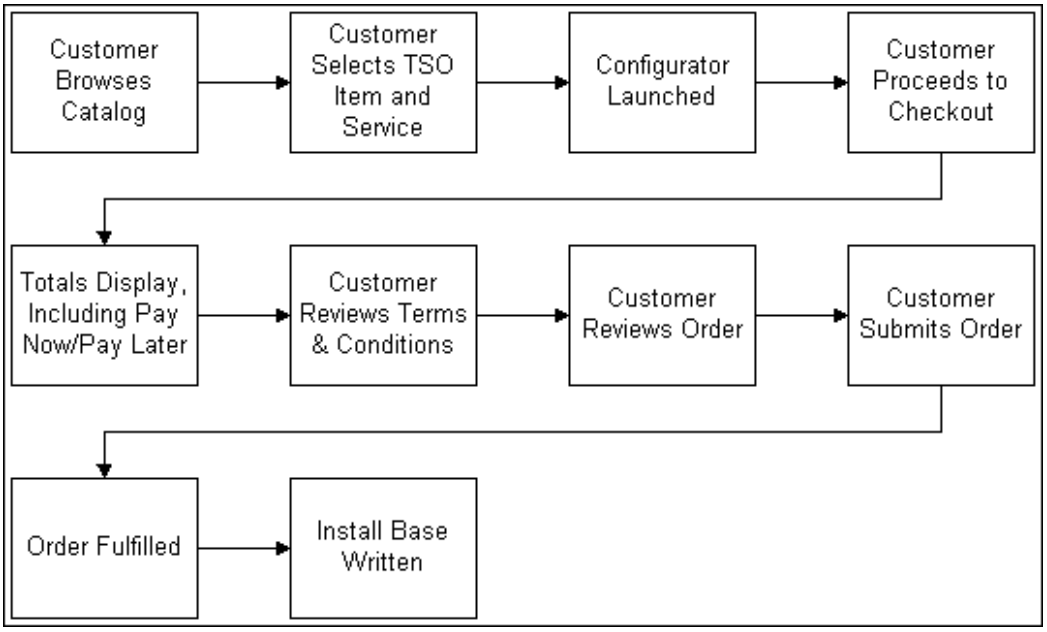
6. The customer proceeds to checkout, where Oracle iStore captures shipping (only if shippable items are present in the cart) and billing information.

7. Oracle iStore displays a preview of the order in the shopping cart, along with prices for shippable items and one-time and recurring charges for the service. The preview informs the customer which amounts he must pay the service provider immediately (Payment Due with Order amount) and which can be paid later. If terms and conditions (T&Cs) functionality has been implemented, Oracle iStore also displays T&Cs pertaining to the items in the cart. The preview page gives the customer the option to make changes to the configuration in the cart or proceed to checkout after accepting the T&Cs.

8. The customer reviews the order, accepts the T&Cs, and places the order, which is ultimately provisioned and fulfilled.

9. After fulfillment, the item instances are written in installed base. In the Oracle iStore Customer Application, the customer can track the order and/or view and reconfigure his installed base.

Oracle iStore Flow for New Configuration



Oracle iStore Flow for Reconfiguration

Note that customers cannot start the reconfiguration process if any of the item instances has a pending order against it. In this case, the system displays an error message.

Prerequisites: A customer has purchased a TSO item instance, the item instance has been provisioned, and the details have been written to the customer's installed base.

1. A customer wishes to revise or reconfigure a TSO item. He accesses his installed base through the My Products tab in Order Tracker in the Oracle iStore Customer Application. For example, the customer may be considering the following changes:

   - Change the service package: A customer who has wireless phone service and currently subscribes to the 500 Anytime Minutes package may wish to change to the 1000 Anytime Minutes package.

   - Change the features of an existing service: The wireless phone service customer may wish to subscribe to some additional services, such as conference calling or international roaming, or may wish to disconnect some existing service, such as caller ID blocking.

   - Change/upgrade physical items associated with the service: The wireless customer may wish to upgrade to a new handset.

   The customer may also consider a change that is a combination of the three options listed above.

2. The customer locates the item and selects the Details/ Reconfigure icon for the item, and then selects Reconfigure.

3. Oracle iStore launches Oracle Configurator, and the customer configures the item instance, selecting the options he wishes to purchase. Note that Oracle iStore provides a separate reconfiguration cart (separate from the active cart) for this purpose. The active cart remains active. The two carts must be kept separate since the reconfiguration cart should not be saved (this would be a problem if the install base changed) or duplicated (there would be a problem with two shopping carts against the same install base instance). The reconfiguration cart can be converted to a quote if the user requests sales representative assistance.

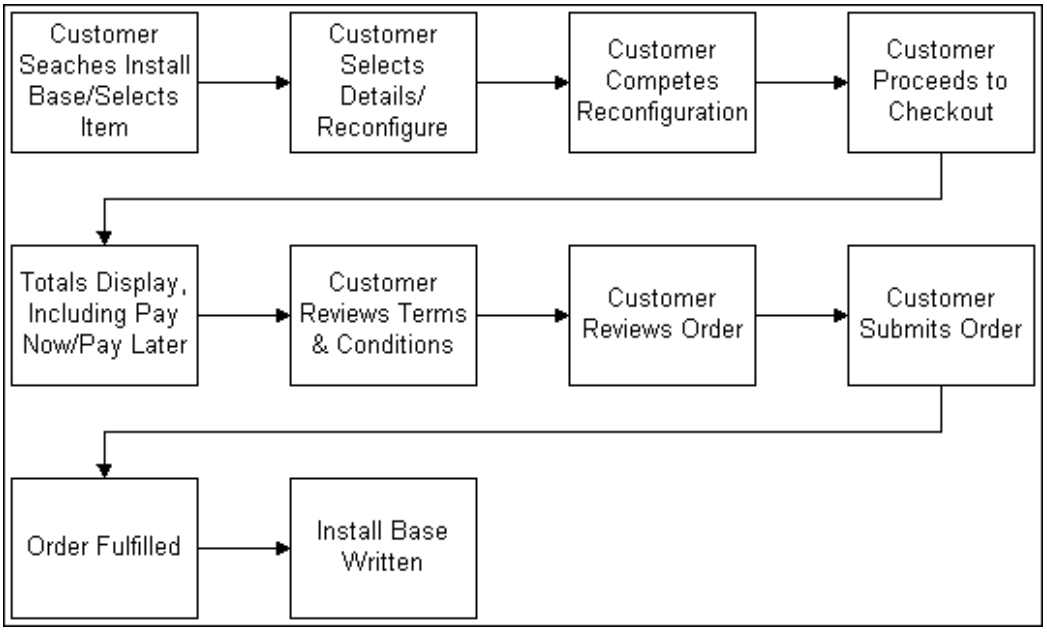
4. Once the changes are complete, the customer clicks Done in the Configurator UI and then agrees to the T&Cs (if T&C functionality is implemented).

5. Oracle iStore displays a summary of the item changes in the Review Changes page. The Review Changes page shows: customer information, billing information, shipping information (if shippable items are included), the reconfigured item instances, and the one-time and recurring prices for the changes, including what the customer must pay now and later. For more information about the behavior of the Review Changes page, see the chapter, Using the Oracle TSO Solution, page 16-1.

6. The customer verifies the changes and submits the order, which is ultimately fulfilled.

Note that if the customer does not submit the changes at this time, then all the changes will be lost, and the customer would need to go through the reconfiguration flow again to reconfigure the items.

Oracle iStore Flow for Reconfiguration



Oracle Contact Center TSO Flows

All the flows in Oracle Contact Center involve user interaction with the customer on the phone. Hence the agent's interaction with the system begins and ends with the customer on the phone.

Oracle Contact Center UI Flow for New Configuration

Following is the Oracle Contact Center Flow for a new configuration.

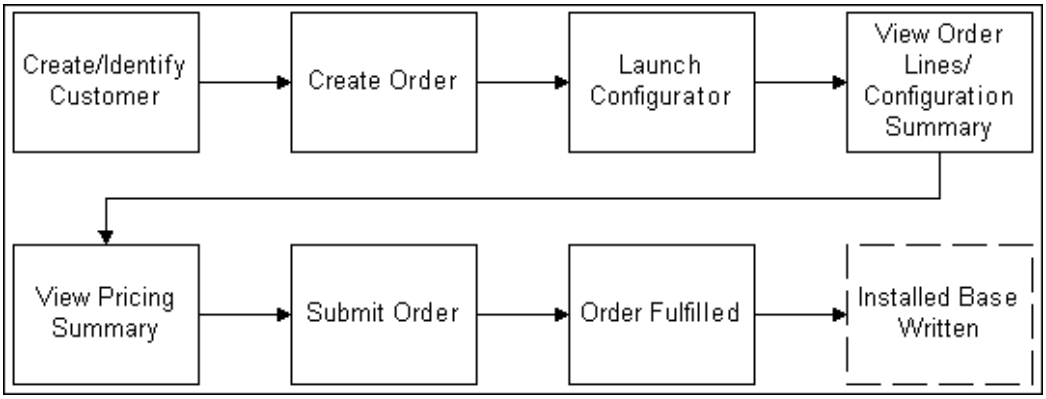
1. The sales agent selects an existing or creates a new customer in the Contact Center header page.
2. The sales agent accesses the Order Information, Order List, or Line Items subtab and creates an order using the customer.
3. The sales agent launches Oracle Configurator and configures the product and services.
4. After the item is configured as required, the configuration window is closed and the agent views the configuration summary in the Order Line Items subtab.
5. The agent views the pricing summary in the Order Summary subtab.
6. The agent submits the order.

7. The order is fulfilled.

8. After provisioning, the details are written to the customer's installed base.

The following figure shows the flow.

Oracle Contact Center UI Flow for New Configuration



Oracle Contact Center UI Flow for Reconfiguration (B2C)

Following is the Oracle Contact Center Flow for a B2C reconfiguration.

1. The sales agent selects an existing customer in the Contact Center header page.

2. The sales agent queries Oracle Installed Base for the products or services to reconfigure.

3. The sales agent launches Oracle Configurator and reconfigures the product and services.

4. After the item is configured as required, the configuration window is closed and the agent views the configuration summary in the Order Line Items subtab.

5. The agent views the pricing summary in the Order Summary subtab.

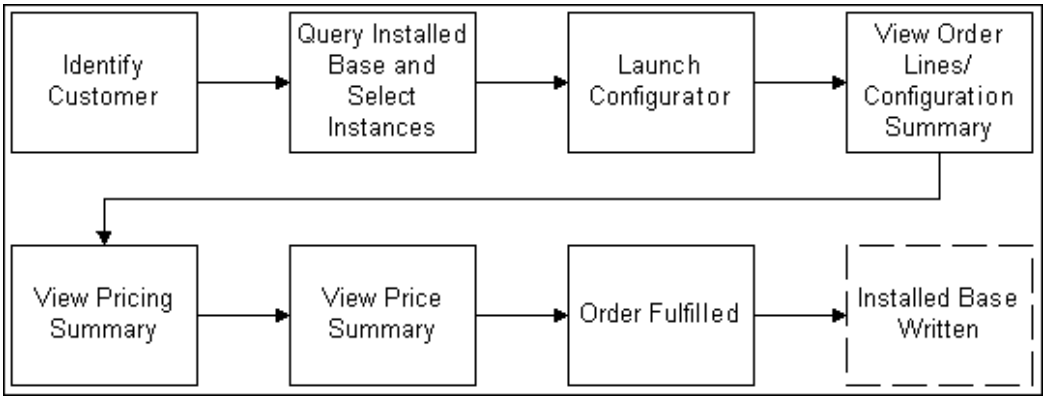
6. The agent submits the order.

7. The order is fulfilled.

8. After provisioning, the details are re-written to the customer's installed base.

The following figure shows the flow.

Oracle Contact Center UI Flow for Reconfiguration (B2C)



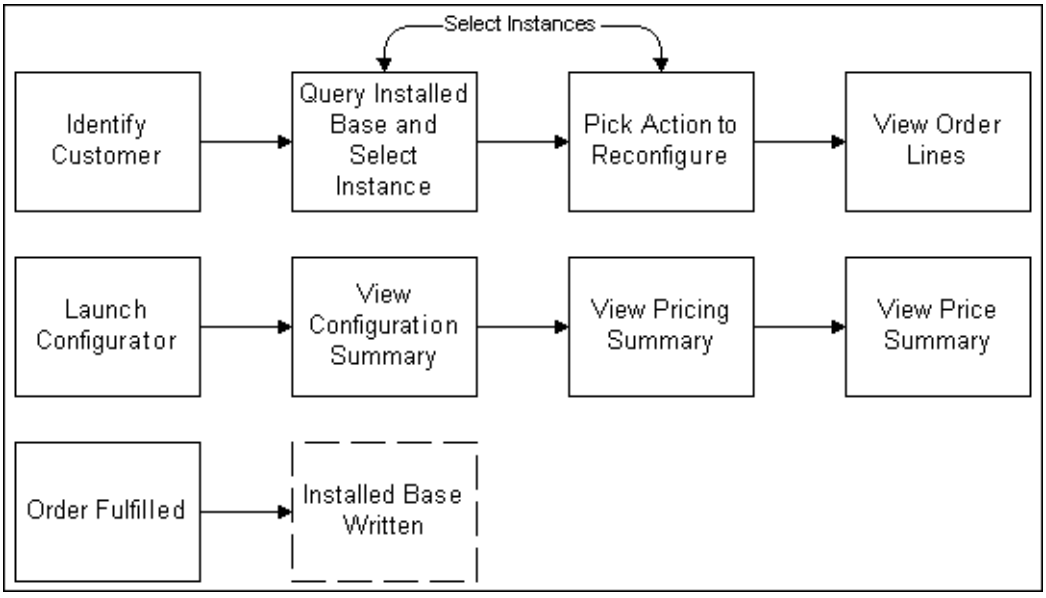
Oracle Contact Center UI Flow for Reconfiguration (B2B)

Following is the Oracle Contact Center Flow for a B2B reconfiguration.

1. The sales agent selects an existing customer in the Contact Center header page.
2. The sales agent queries Oracle Installed Base for the products or services to reconfigure.
3. The agent views the Selected Instances subtab and repeats the query-and-pick process until all of the required items are found.
4. The sales agent launches Oracle Configurator and reconfigures the selected product and services.
5. After the item is configured as required, the configuration window is closed and the agent views the configuration summary in the Order Line Items subtab.
6. The agent views the pricing summary in the Order Summary subtab.
7. The agent submits the order.
8. The order is fulfilled.
9. After provisioning, the details are re-written to the customer's installed base.

The following figure shows the flow.

Oracle Contact Center UI Flow for Reconfiguration (B2B)



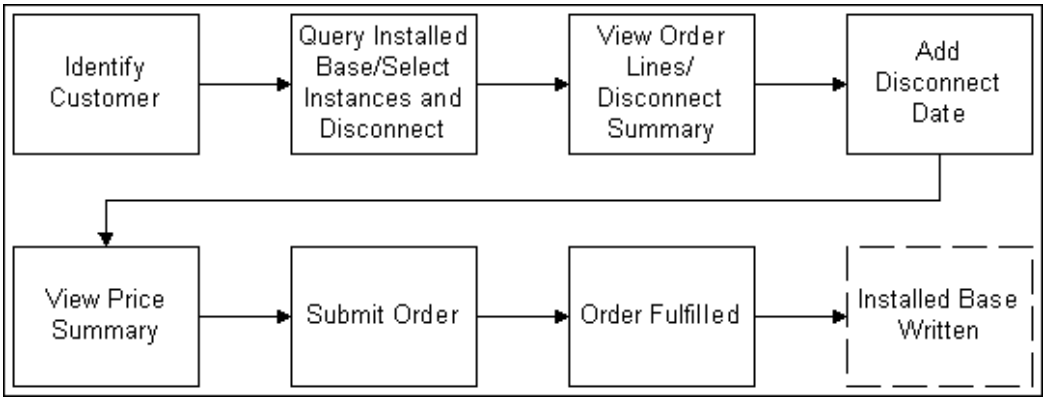
Oracle Contact Center UI Flow for Disconnect (B2C)

Following is the Oracle Contact Center Flow for a B2C disconnect.

1. The sales agent selects an existing customer in the Contact Center header page.
2. The sales agent queries Oracle Installed Base for the products or services to disconnect.
3. The sales agent views the Order Management Line Items subtab to disconnect the item.
4. The agent access the Order Management Order Information subtab to add a disconnect date.
5. The agent views the pricing summary in the Order Summary subtab.
6. The agent submits the order.
7. The order is fulfilled.
8. After provisioning, the details are re-written to the customer's installed base.

The following figure shows the flow.

Oracle Contact Center UI Flow for Disconnect (B2C)



Oracle Advanced Pricing Functionality

Oracle Advanced Pricing provides pricing infrastructure and information to the Oracle TSO solution. Oracle Advanced Pricing supports recurring and one-time charges, as well as discounts on these charges.

Oracle Service Fulfillment Manager Functionality

Oracle Service Fulfillment Manager provisions the telecommunication services that customers order; it does this by performing the following actions:

- Capturing a service order request
- Validating the order
- Analyzing the order
- Fulfilling the order
- Completing the order

If necessary, Oracle Service Fulfillment Manager also manages order fallout.

Refer to *Oracle Service Fulfillment Manager Concepts and Procedures* for a description of the provisioning process. The provisioning functions are the same for the Oracle TSO solution as they are for any other service order request that Oracle Service Fulfillment Manager receives. However, Oracle Service Fulfillment Manager requires some specific setup steps to receive a configured order from Oracle Order Management and to retrieve and reconfigure extended attributes from the Oracle Installed Base Transaction Details window.

For more information about setting up Oracle Service Fulfillment Manager, see the

chapter, Set Up Oracle Service Fulfillment Manager, page 15-1, within this guide, and the *Oracle Service Fulfillment Manager Implementation Guide*.

Oracle Telecommunications Billing Integrator Functionality

Oracle Telecommunications Billing Integrator (TBI) is an interface between the Oracle E-Business Suite applications and external telecommunications billing applications. Information on customers, accounts, products and sales orders are replicated and kept in sync to ensure that bills are computed accurately.

In summary, Oracle TBI provides the following:

- **Exchange information:** Oracle TBI enables the exchange of information between Oracle E-Business Suite and integrated external billing applications. Oracle TBI publishes details on accounts, products, and orders to external billing applications, and the Oracle E-Business Suite can receive acknowledgements for the published messages.

- **Integrate with multiple billing systems:** Oracle TBI can integrate with multiple heterogeneous billing applications simultaneously.

- **Track status:** Oracle TBI integrates with the Oracle Collaboration History module to monitor communication between Oracle E-Business Suite and the billing applications.

- Display Bill Summary and Bill Image: Oracle TBI provides access to the bill summary data through Oracle Contact Center. The Bill Summary UI displays the bill summary for a single account number. Sales agents can view the bill summary from the Bill Summary option in Oracle Contact Center.

- **Leverage business event system:** The business event system is a real-time notification system which notifies Oracle TBI when an account or sales order is created or updated.

Refer to the *Oracle Telecommunications Billing Integrator Implementation Guide* for further details.

4

Implementing the Oracle TSO Solution

This chapter covers the following topics:

- Overview of Implementing the Oracle TSO Solution
- Mandatory Dependencies for the Oracle TSO Solution
- Optional Integrations
- Oracle TSO Setup Sequence
- Implementation Flows for Specific Functionality

Overview of Implementing the Oracle TSO Solution

This chapter includes setup information that is specific to the Oracle Telecommunications Service Ordering (TSO) solution.

The Oracle TSO functionality leverages the following Oracle Applications:

- Oracle Accounts Receivable
- Oracle Advanced Pricing
- Oracle Advanced Product Catalog
- Oracle Configurator
- Oracle Contact Center
- Oracle Bills of Material
- Oracle iStore
- Oracle Installed Base
- Oracle Inventory

- Oracle Order Management
- Oracle Quoting
- Oracle Service Fulfillment Manager
- Oracle Telecommunications Billing Integrator

The mandatory applications provide core TSO functionality (see "Mandatory Dependencies for the Oracle TSO Solution", below). Others are required only if the related functionality is desired. TSO-specific implementation information for these applications is located within this guide. Additional information on setting up core functionality of these applications can be found in the relevant documentation for the specific application.

Note: Due to the limited or generic nature of the functionality provided by some of these applications, this document does not contain separate implementation chapters on Oracle Accounts Receivable, Oracle Advanced Product Catalog, Oracle Telecommunications Billing Integrator, or Oracle Trading Community Architecture.

Mandatory Dependencies for the Oracle TSO Solution

Oracle TSO functionality requires integration with several other Oracle applications. Therefore, you must set up the required dependencies. The following table and sections outline the dependencies.

Note: This document provides additional setup information that is not included in the implementation guides of the Oracle applications that are required to support the TSO solution. You should use this document in conjunction with the implementation guides of each of these Oracle applications.

The following table lists the TSO functionality provided by the Oracle applications involved in the TSO solution.

Oracle TSO Functionality and Application Dependencies

Application	Functional Dependency Related to TSO
Oracle Accounts Receivable	Payment Due with Order functionality
Oracle Advanced Pricing	Recurring charges; provides an attribute for periodicity
Oracle Advanced Product Catalog	Recurring charges

Application	Functional Dependency Related to TSO
Oracle Bills of Material	Required for Oracle Configurator
Oracle Configurator	Configurable items; container models; partial reconfiguration of network items; computation of configuration deltas; persistence of attributes and Line Types; tangible item in a network container
Oracle Installed Base	For tangible items in a network container, provides an API to prevent reconfiguration of "locked" instances; stores purchased products
Oracle Inventory	Recurring charges; Payment Due with Order; provides a periodicity UOM class and UOM codes
Oracle Order Capture schema	Payment Due with Order; recurring charges; defaults the charge periodicity code on the quote line from Inventory; provides support for periodicity-based pricing; provides a source rule to derive the periodicity code
Oracle Order Management	Capable of capturing, booking, and provisioning orders; Payment Due with Order; recurring charges; tangible item in a network container; provides profile option to specify the periodicity UOM class; provides a system parameter to enable recurring charges; API dependency for Order Tracker and e-mail notifications
Oracle Trading Community Architecture	Stores information about customers and their interconnected relationships

For information about implementing these Oracle applications, see the respective product implementation guides.

Payment Due with Order Dependencies

For Payment Due with Order functionality, the Oracle TSO solution has a dependency on Oracle Accounts Receivable (AR) via Oracle Order Management. AR provides an API that calculates the pay now amount. Oracle Order Capture schema handles pay now subtotals, pay now taxes, pay now shipping/handling, and pay now grand totals. Oracle Inventory provides the setup UI for item-level payment terms for Payment Due

with Order. In Oracle iStore, Oracle Order Capture defaults item-level payment terms from the Inventory Item Master during the add-to-cart process.

Recurring Charges Dependencies

For recurring charges, Oracle Order Capture schema handles the following:

- Defaults the charge periodicity code on the quote line from the Oracle Inventory Item Master
- Provides support for periodicity-based pricing
- Provides a source rule to derive the periodicity code

Oracle Advanced Product Catalog seeds the attribute that stores periodicity of TSO items.

Below is a list of the dependencies on Oracle Order Management:

- Provides the profile option to specify the periodicity UOM class. When selecting a periodicity while creating an item, the periodicity needs to belong to the UOM class.
- Provides a system parameter to enable recurring charges.
- API dependency for Order Tracker and e-mail notifications

Oracle Advanced Pricing provides a pricing attribute for periodicity. Oracle Inventory provides a periodicity UOM class and UOM codes.

Tangible Items in Network Container Dependencies

During the RMA flow, Oracle Installed Base calls an Oracle Configurator procedure to remove tangible items from the configuration. The configuration is validated after the removal of the tangible item.

If a Model includes non-shippable items and at least one shippable (tangible) item, when the tangible item is shipped, Oracle Order Management ensures that the shipped quantity is not updated for all the other lines in the Container Model.

All ordering applications have a dependency on Oracle Order Management and Oracle Configurator for this functionality with the following limitations:

- Only shipping (ADD) and discontinue (DISCONNECT) flows is supported for tangible items within a container.
- Only shipping of standard items and kits within a container is supported; PTO and ATO models are not supported.
- The quantity of the root model (MI, PTO model) will always be 1, and will be

non-shippable.

- The quantity of tangible items cannot be more than 1.
- Replacement flows that transfer warranties by using the Oracle Installed Base "Replace By" relationship are not supported.
- The use of extended attributes for tangible items is *not* supported.
- No connectors are allowed from/to tangible items within a container.
- Removal of tangible items from a configuration will result in deleting the item in Oracle Configurator and de-coupling the item from the configuration in Oracle Installed Base. In order to return the item that was removed from the configuration, the standard RMA process needs to be followed.

Optional Integrations

The following Oracle applications can be integrated to use the Oracle TSO solution:

- Oracle Contact Center (TeleService)
- Oracle iStore
- Oracle Quoting
- Oracle Service Fulfillment Manager
- Oracle Telecommunications Billing Integrator

For information about the functionality related to TSO within these products, see the chapters, Oracle TSO Functionality, page 3-1 and Using the Oracle TSO Solution, page 16-1, as well as the relevant setup chapters with in this guide.

Oracle TSO Setup Sequence

The setup flow below shows a typical Oracle application setup sequence for the Oracle TSO Solution.

Set up products in the following order:

1. Oracle Inventory
2. Oracle Advanced Product Catalog
3. Oracle Bills of Material and Oracle Installed Base

4. Oracle Configurator
5. Oracle Order Management (workflows only)
6. Oracle Service Fulfillment Manager
7. Oracle Advanced Pricing
8. Oracle Quoting, Oracle Contact Center, Oracle Order Management (except workflows), and Oracle iStore

Implementation Flows for Specific Functionality

This section contains high-level implementation flows for recurring charges and Payment Due with Order functionality.

Recurring Charges Implementation Flow

Following are the product-specific implementation flows for setting up recurring charges in the Oracle TSO solution.

Oracle Advanced Pricing Setups

For recurring charges setup in Oracle Advanced Pricing, attach the Charge Periodicity pricing attribute to a price list line or a modifier line.

Oracle Inventory Setups

For recurring charges setup in Oracle Inventory, set up recurring charges items with a value (e.g., Monthly) in the in the Charge Periodicity field.

Oracle Order Management Setups

Following are the Oracle Order Management setups for recurring charges:

- Set the system parameter, Enable Recurring Charges.
- Set the profile option, OM: UOM Class for Charge Periodicity, to Period.
- Modify the Order Management UI to show charge periodicity.

Oracle iStore Setups

For recurring charges setup in Oracle iStore, set the profile option, IBE: Enable Recurring Charges, to Yes.

Oracle Quoting Setups

For recurring charges setup in Oracle Quoting, expose the Charge Periodicity column from the Products subtab.

Payment Due with Order Implementation Flow

Following are the product-specific implementation flows for setting up Payment Due with Order in the Oracle TSO solution.

Oracle Inventory Setups

Following are the Oracle Inventory setups for Payment Due with Order:

- For each item that will have a Payment Due with Order amount associated with it:
  - Set the Payment Terms field to Pay Now.
  - Enable the Invoiceable Item and Invoice Enabled check boxes.

Oracle Order Management Setups

Following are the Oracle Order Management setups for Payment Due with Order:

- Set the Order Management system parameter, Installment Options.
- Set up payment term or terms for Payment Due with Order amount.
- Set up defaulting rules automatically default line-level payment terms and payment type.
- Set up a credit checking rule to specify at ordering for the transaction type being used for the order.
- Assign the following workflow processes to the line transaction types that will be used: Line Flow - Generic, Bill Only with Payment Assurance and Line Flow - Generic, with Payment Assurance.
- Set up taxes to be calculated at the entry point, not inclusive in the lines.
- Expose the following fields in the Order Management UI: Initial Due Balance, Due with Order, Initial Due Subtotal, Tax, Charges, and Total.

Oracle iStore Setups

For Payment Due with Order setup in Oracle iStore, set the profile option, IBE: Enable Pay Now, to Yes.

Oracle Quoting Setups

For Payment Due with Order setup in Oracle Quoting, display UI elements specific to the Payment Due with Order amount.

5

Set Up Oracle Order Management

This chapter covers the following topics:

- Overview of Set Up Oracle Order Management Chapter
- Map Actions and Line Types
- Enable Recurring Charges
- Enable Payment Due with Order
- Set Profile Options
- Set Up Additional Line Types
- Create SFM-Enabled Order Header and Line Workflows
- Remove Fulfill Node in Line Workflow
- Design Tips

Overview of Set Up Oracle Order Management Chapter

This chapter describes setup tasks that you must perform in Oracle Order Management (OM) that are specific to the Oracle Telecommunications Service Ordering (TSO) solution.

Before You Begin

Among the Oracle Applications that are directly a part of the Oracle TSO solution, there are no dependencies to setting up Oracle Order Management. It is recommended that you set up Oracle Order Management first.

It is assumed, however, that Oracle Accounts Receivable has been implemented, or that your implementation has access to the APIs provided by Oracle Accounts Receivable.

Setup Checklist

Following is the setup checklist for implementing TSO-specific functionality for Oracle Order Management:

Oracle Order Management-TSO Setup Checklist

Setup Step	Required/Optional
Implement Oracle Order Management	Required
Map Actions and Line Types	Required
Enable Recurring Charges	Required
Enable Payment Due with Order	Optional
Set Profile Options	Required
Assign Workflows to Transaction Types	Required
Set Up Additional Line Types	Optional
Create SFM-Enabled Order Header and Line Workflows	Optional
Remove the Fulfill Node in the OM Line Workflow	Optional

For more information on setting up and using Oracle Order Management, refer to:

- *Oracle Order Management Implementation Manual*
- *Oracle Order Management User's Guide*

Map Actions and Line Types

As an implementation requirement, map all Actions and Line Types associated with the Oracle Configurator Line Type Configurator Extension to the Order Line Type in Oracle Order Management Transaction Types setup.

Enable Recurring Charges

Products with one-time and recurring charges are entered on separate transaction lines in the sales order. A folder-enabled column, labelled Charge Periodicity Code, is available on the order line to store and display the charge periodicity. *Charge periodicity* is the interval by which the price for a recurring charge has been set up on a price list (e.g., Monthly, Quarterly, Yearly). The charge periodicity indicates how often the customer account is charged. Order lines with a charge periodicity are considered recurring charge lines. Order lines without a charge periodicity are considered one-time charges. When ordering an item with a recurring charge, the charge periodicity is defaulted on the order line from the item master using the defaulting rules. For one-time charges (i.e., lines without a charge periodicity), the charge periodicity column displays as blank.

Setup of recurring charges in Oracle Order Management involves the following steps:

1. Set an Order Management profile option.
2. Set an Order Management system parameter.
3. Specify charge periodicity for items with recurring charges.
4. Modify UI to show charge periodicity.

See the sections below for more information.

Set Profile Option

Set the profile option, OM: UOM Class for Charge Periodicity, to Period. See the section, "Set Profile Options", later in this chapter for more details.

Set Parameter

The Oracle Order Management system parameter, Enable Recurring Charges, controls the recurring charge feature and cannot be disabled once the parameter is set/enabled. Set the parameter to Yes. Consult the *Oracle Order Management Implementation Manual* for instructions detailing how to set system parameters.

Specify Charge Periodicity for TSO Items

For each TSO item requiring the use of recurring charges, specify the periodicity of the item. For these types of items, the periodicity is the unit of measure.

Steps

1. As Order Management Super User, navigate to the Master Items window and query the item.

2. In the Order Management tab for the item, specify a value of Monthly (or another value, as appropriate) in the Charge Periodicity field.

Modify UI to Display Charge Periodicity

Modify the UI to display charge periodicity elements:

- Add the Charge Periodicity field to the folder in the header block and line blocks in the Sales Order and Quick Sales Order forms (Oracle Order Management forms).
- Add the Charge Periodicity field to the line region in the Order Details page using personalization (Order Information Portal).

For more information, refer to the *Oracle Order Management Implementation Manual* and the *Oracle Application Framework Personalization Guide*.

Enable Payment Due with Order

This section covers the high-level steps to enable Payment Due with Order functionality. Note the following:

- A line is considered a pay now line if the pay now portion of the line is greater and 0 (zero).
- The pay now payment term must have at least one installment with 0 (zero) days due.
- A pay now portion of an order line is a sum of all installments with 0 (zero) days due.
- The pay later payment term must have all installments with days due greater than 0 (zero).

Setting up Payment Due with Order involves defining business logic that will determine the "pay now" amount on an order. The following tasks must be performed:

1. Set an Order Management system parameter.
2. Set up payment term or terms.
3. Set up defaulting rules.
4. Set up credit checking rule.
5. Set item attributes.
6. Assign line workflows

7. Set up taxes.

8. Expose fields in the Order Management UI.

Set System Parameter

Set the Oracle Order Management system parameter, Installment Options, to Enable Pay Now. This system parameter was previously called Authorize First Installment Only. The system parameter cannot be changed if it is already set to Enable Pay Now or Authorize First Installment. The following table shows the implications of the various settings of the parameter.

Installment Options System Parameter Settings and Implications

Parameter Value	Implication
Enable Pay Now	Enables Payment Due with Order functionality.
Authorize First Installment	Only the first installment of a payment is authorized. The amount authorized will be total of the first installment less down payment, if applicable
None	Does not enable the Payment Due With Order and Authorize First Installment functionality

Set Up Payment Term

Set up a payment term in for Payment Due with Order. Payment terms that have one or more installments with Days Due set to zero will be used to identify the Payment Due with Order order lines.

Steps

1. As Order Management Super User, navigate to Setup, Orders, Payment Terms.

2. Enter a name for the payment term (e.g., Pay Now).

3. In order for the payment term to be a "pay now" payment term, ensure that no installment is allowed. You specify this by entering 0 (zero) in the Days Due field in the Payment Schedule region).

Set Up Defaulting Rules

The item-level payment term determines if an order line is included in the Payment Due with Order amount. If the line is to be included in the Payment Due with Order amount, then the payment term should have Days Due set to zero. Implementers can set up a defaulting rule to automatically specify the line level payment terms based on business rules. Additionally, the sold-to customer type can be used to default different payment terms for B2B customers versus B2C customers.

Steps

1. As Order Management Super User, navigate to Setup, Rules.
2. Choose the Defaulting Application Name as Order Management.
3. Query for the Order Line entity and Payment Term attribute.
4. Set the Defaulting sourcing rule to Related Record.
5. Select the defaulting rules button.
6. Set the default source/value, related object to Inventory Item and related record to Payment Term.

Implementers also should set up defaulting rules to default Credit Card as the payment type. The payment type, Credit Card, is seeded so that processing can be immediate (non-deferred). To ensure that credit card is authorized at booking, the Defer Payment Processing Flag should be set to No.

Set Up Credit Check Rule

Specify the credit check rule at ordering for the transaction type being used for the order, so that credit card can get authorized.

Steps

1. As Order Management Super User, navigate to Setup, Credit, Define Credit Check Rules.
2. Either create a credit check rule or use an existing one and specify this credit check rule in the transaction type for the order.

Set Up Items

For each TSO inventory item that will have a Payment Due with Order amount, set the Payment Terms field to the Pay Now payment term. Also, enable the Invoiceable Item and Invoice Enabled check boxes.

Assign Line Workflows to Transaction Types

Assign the following workflow processes to the line transaction types that will be used:

- Line Flow - Generic, Bill Only with Payment Assurance
- Line Flow - Generic, with Payment Assurance

Consult the *Oracle Order Management Implementation Manual* for instructions.

Set Up Taxes

To include tax in the pay now portion, set up the tax to be calculated at the entry point. And tax should not be set up to be "inclusive" in the lines.

Modify Header Folders

Using folder modification technology available in Oracle Forms, implementers can modify the UI to display Payment Due with Order amount elements. Following are the suggested modifications for Payment Due with Order functionality. Add the First Installment fields and the Due with Order checkbox to the header folders in the Payments and Line Payments forms.

- Quick Sales Orders form: Add the Initial Due Balance field in the order folder under the Main tab
- Payments form (available from the Quick Sales Orders form):
  - Add the Due with Order checkbox.
  - Add the Initial Due Subtotal, Tax, Charges, and Total fields.

Set Profile Options

Set the profile options listed in this section.

OM: Use Configurator

This profile option indicates which Oracle application software is launched to enter configuration information when selecting the Configurator button from the Sales Order window. Set to Yes to integrate with Oracle Configurator in the Oracle TSO Solution. For more information, see the *Oracle Order Management Implementation Manual*.

OM: UOM Class for Charge Periodicity

Set the profile option, OM: UOM Class for Charge Periodicity, to Period. This profile

option specifies which unit of measure (UOM) class stores the available charge periodicities.

OM: Sales Order Form: Cascade Header Changes to Line

When a header attribute is updated and the lines are entered, any header changes are cascaded down to the lines. In the reconfiguration flow, the header information will be derived from the item instance that was selected, together with any Oracle Order Management defaulting rules. Because the header information defaults from the information available for the selected item instance in the installed base, there is a high possibility that the header information will need to be updated after the order is created. For this reason, it is necessary to provide an option to cascade many of the changes to the order header to the respective fields on the order line of the order. The profile option, OM: Sales Order Form: Cascade Header Changes to Line, determines the cascading behavior. Following are the profile option settings:

- Automatic: The system cascades the header attributes to lines whenever a header attribute is cascaded
- Askme: The user is given a choice to cascade or not
- Manual: The user must manually change the values for the attributes on the lines; the system doesn't cascade the changes from header to lines.

Set Up Additional Line Types

You can set up additional customized Line Types for reconfigurations.

For instructions on how to set up additional Line Types, refer to the steps for defining Transaction Types in the *Oracle Order Management Implementation Manual*. For information on setting up the Line Type that ensures that orders pass into Oracle Service Fulfillment Manager, see the *Oracle Service Fulfillment Manager Implementation Guide*.

For more information on related TSO setup for Line Types, see the section, "Line Type Configurator Extension", in the chapter, Set Up Oracle Configurator Extensions, page 11-1.

Create SFM-Enabled Order Header and Line Workflows

The Fulfill activity is an OM Order Line activity that calls the PL/SQL procedure named OE_FULFILL_WF._FULFILLMENT to perform fulfillment for an order line. The Fulfill activity contains the following attributes:

1. Fulfillment Activity Name
2. Completion Result

3. Inbound Fulfillment Activity Name

4. Inbound Completion Result

The Fulfill activity coordinates the fulfillment of lines so that they progress together to invoicing. It applies to Models and kits (shippable and non-shippable lines), fulfillment sets, and service lines attached to goods. The Fulfill activity also derives and stores additional data needed for Oracle Daily Business Intelligence (DBI) and service lines. The Fulfill activity is contained in the seeded data file, oexwford.wft, and is associated with the OM Order Line item type. Note that in WF, Fulfill is an "activity" and not a " node".

This step is required only if you are using Oracle Service Fulfillment Manager (SFM).

To enable the flow of order from Oracle Order Management to Oracle Service Fulfillment Manager, you must make some required customizations to Oracle Order Management's Workflow processes. This enables fulfillment of provisionable items from Oracle Order Management by way of SFM. For more information, see the chapter, Set Up Oracle Service Fulfillment Manager.

Remove Fulfill Node in Line Workflow

Removing the Fulfill node in the Oracle Order Management line workflow is optional.

The Fulfill node does the following:

- Performs a gating activity for Components of a configuration.
- Ensures that all the lines in a configuration are fulfilled together.

As a consequence of the gating activity, the update of Oracle Installed Base occurs only after fulfillment of all the Components of a configuration. In the business-to-business communications world, especially in the data services like Frame Relay, provisioning of Components may occur days or months apart from each other.

Because of the gating activity of the Fulfill node in the Order Management line workflow, the updating of installed base does not occur until provisioning of all the Components of the configuration occur. For example, this might be three months after provisioning the first Component. This can result in the installed base not accurately reflecting all of the customer's current services. To avoid this issue, it is recommended that you replace the Fulfill node with one that stamps fulfilled flag, the fulfillment date, and the fulfillment quantity for each order line.

Oracle Installed Base does not process an order line without a fulfillment date. It is essential that if you remove the Fulfill node, you replace it with a node that stamps the fulfilled flag, the fulfillment date, and the fulfilled quantity of each order line.

You should replace the Fulfill node only if your implementation requires that the order lines of a Container Model are fulfilled independently of each other.

For more information, see:

- *Oracle Order Management Implementation Manual*
- *Oracle Order Management User's Guide*

Design Tips

Keep the following design tips in mind as you implement and use the Oracle TSO Solution with Oracle Order Management:

- You cannot reconfigure Container Models after booking. You can always reconfigure Container Models when the order is in the negotiation phase. If the order is in the Entered state, you can reconfigure Container Models only if the line workflow that is attached to the line types is the same. For more information, see the appendix, Assumptions and Restrictions, page A-1.
- Payment terms that have one or more installments with days due equal to zero are used to mark pay now order lines.
- A payment term may consist of pay now and pay later portions.
- Implementers should define their defaulting rules to assign payment terms to order lines. The seeded Party Type is a defaulting attribute that can be used in the condition template, since it is common that the party type of the customer (B2B or B2C) determines if an amount on an order line needs to be paid now or not. Party Type is available on the order header and the order line.
- The warehouse for a recurring charge line should not be changed during the reconfiguration flow. Thus, if a recurring charge line had warehouse W1 during the initial configuration, then it should have W1 during the reconfiguration process.
- The workflow needs to be modified for the disconnect flow of a shippable line, so that it doesn't interface to shipping.

6

Set Up Oracle Inventory and the Item Master

This chapter covers the following topics:

- Overview of Set Up Oracle Inventory and the Item Master
- Set Up Items
- Set Up Periodicity for Recurring Charges
- Create Container Model
- Set Items as Provisionable
- Set Up Link Items

Overview of Set Up Oracle Inventory and the Item Master

This chapter describes setup tasks for the Oracle Inventory Item Master (which is also accessible through other applications; e.g., Oracle Order Management) and Oracle Inventory items that are specific to the Oracle Telecommunications Service Ordering (TSO) solution.

Setup Checklist

Following is the setup checklist for implementing TSO-specific functionality for inventory items.

Oracle Inventory Items-TSO Setup Checklist

Setup Step	Required/Optional
Implement Oracle Inventory	Required
Set Up Items	Required

Setup Step	Required/Optional
Set Up Periodicity for Recurring Charges	Required
Create Container Model	Required
Set Items as Provisionable	Optional
Set Up Link Items	Optional

For more information on setting up items, refer to the *Oracle Inventory User's Guide*.

Set Up Items

Implementers can define TSO items with various flags that determine the functionality available to the item. The TSO-related flags and settings are summarized below; some of the flags' setups are expanded in subsequent sections.

- **Install Base Trackable:** This flag allows the tracking of the life cycle of the product in Oracle Installed Base. For more information, see the section, "Specify Installed Base Tracking", below.
- **Payment Terms:** For each TSO inventory item that requires payment with the order, set the Payment Terms field to Pay Now.
- **Invoicing:** Enable the Invoiceable Item and Invoice Enabled checkboxes for each item.
- **Instance Class Network Link:** This value of the Instance Class attribute is required only if you use the selected inventory item as a network link. For more information, see the section, "Set Up an Item of Type Link", below.
- **Provisionable:** This is required if you are going to provision or electronically fulfill the item. For more information, see the section, "Set an Item as Provisionable", below.
- **Container Model:** A Container Model is a model that you use to contain TSO products that you can reconfigure. For more information, see the section, "Create Container Model", below.
- **Periodicity:** For each recurring item, specify a period (e.g., Monthly) in the Charge Periodicity LOV. See "Set Up Periodicity for Recurring Charges", below for more information.

Set Up Periodicity for Recurring Charges

To support recurring charges in the TSO solution, Oracle Inventory seeds a unit of measure (UOM) class labelled, Period, with relevant UOMs to support the various periods that recurring charges might be billed as. Implementers should set the Period UOM to the period of the recurring charge. The following table shows the seeded values for the Period UOM class:

Seeded Values Period Class

Code	Description
DAY	Daily (base unit)
WK	Weekly
BWK	Bi-Weekly
MTH	Monthly
QTR	Quarterly
HYR	Half Yearly
YR	Yearly

The default UOM class for recurring charges is derived from the Oracle Order Management profile option, OM: UOM Class for Charge Periodicity. For more information, see the section, "Set Pro□le Options", in the chapter, Set Up Oracle Order Management, page 5-1.

In addition to the UOM class, for each item that will have a recurring charge associated, implementers should set the Charge Periodicity field to the recurring charge period; the field is available in the Item Master form. For more information, see the section, "Specify Periodicity for TSO Items" in the chapter, Set Up Oracle Order Management, page 5-1.

> **Important:** If before an upgrade, the system already had a UOM class named Period, or one or more UOM codes that are in conflict with the seed data, then seed data will not be created and users will not be able to view the seeded UOM Class or codes. Before performing an upgrade, ensure that there is no existing UOM class named Period and that there are no UOM codes with any of the seeded values listed in the table above.

Create Container Model

To create a Container Model, create a model that supports updates to its installed configurable Components using the following procedures:

- "Specify Container Model", below, to indicate that the item representing your top-level BOM Model is a Container Model.
- "Specify Installed Base Tracking", below, to indicate whether or not you want to track each Oracle Inventory item that is a Component of the Container Model.

Specify Container Model

You need to specify that the item representing your top level BOM Model, such as the item representing the network, is a Container Model.

In this procedure, log in to Oracle Forms with Oracle Inventory responsibility and navigate to the Master Item window.

Prerequisite

Review "Container Model Settings and Structure" in the chapter, Set Up Configurator and Customize the Solution, very carefully before defining a Container Model in Oracle applications.

Steps

1. In the Master Item window, query the item or define it, if it does not already exist.
2. Select the Bills of Material tab.
3. On the Bills of Material tab, set the Configuration Model Type to Container.

   Note: The default Configuration Model Type is Standard; this value means only that the item is not a Container Model.
4. Select the Service tab.
5. On the Service tab, verify that the Installed Base Tracking checkbox is not selected.

Note: If a Container Model is marked as trackable, an error occurs when you generate the Active Model in Oracle Configurator Developer.

1. Click the Order Management tab.
2. On the Order Management tab, select the Pick Components checkbox. This indicates that the Container Model is a PTO (pick-to-order) BOM Model. If a Container Model is not a PTO Model, Oracle Configurator Developer displays an error when you import it into the Oracle Configurator schema.

3. Verify that both the Ship Model Complete and Shippable checkboxes are *not* selected.

4. Specify whether or not you want to track each Oracle Inventory item that is a Component of the Container Model. For more information, see: the section, "Specify Installed Base Tracking", below.

5. Save your work.

Specify Installed Base Tracking

After you have specified that the item representing your top level BOM Model is a Container Model, you need to indicate whether or not you want to track, via Oracle Installed Base, each Oracle Inventory item that is a Component of the container BOM Model. Do this for each BOM Model, BOM Option Class, and BOM Standard Item to track in Oracle Installed Base.

Note: The item representing your top-level BOM Model item should *not* have the selected Installed Base Tracking option. Only items to track in Oracle Installed Base should have the Installed Base Tracking option selected.

In this procedure, log in to Oracle Forms with Oracle Inventory responsibility and navigate to the Master Item window.

Steps

1. Query the item or define it if it does not exist.

2. Select the Service tab.

3. On the Service tab, select both the Provisionable and the Installed Base Tracking checkboxes.

4. If you use the item only to link installed Components and you do not usually associate the item with a location, set the Instance Class to Link.

   For example, a Permanent Virtual Circuit (PVC) does not have a specific location, but the PVC connects two Ports in a network configuration model. Upon installation of the item, Oracle Installed Base derives the location from connected the Components.

5. If you wish to track a tangible item, mark it as Shippable, Inventory Transactable, and Serializable.

6. Verify that the Container Model's structure does not violate any of the requirements. For more information, see the section, "Container Model Settings and Structure", in the chapter, Set Up Oracle Configurator and Customize the Solution.

7. Save your work.

Set Items as Provisionable

If you are using Oracle Service Fulfillment Manager, you must specify as provisionable the items that require provisioning in Oracle Service Fulfillment Manager by selecting the Enable Provisioning checkbox for the item.

If you are not using Oracle Service Fulfillment Manager, then these procedures are optional.

In this procedure, log in to Oracle Forms with Oracle Inventory responsibility and navigate to the Master Item window.

Steps

1. In Oracle Inventory, query the item, or define it if it does not yet exist.
2. In the Master Item window, select the Service tab.
3. In the lower left of the tab, select the Provisionable checkbox.
4. Save your work.

Set Up Link Items

You must specify items as the type, Link, only if you use the selected inventory item as a network link. You must set up a configured link item with the Installed Base Instance class = Link.

Example

Suppose ITEM_C is an item with the IB_ITEM_INSTANCE_CLASS set to Link. If ITEM_C is connected to ITEM_A in location A and ITEM_B in location B, then a search for ITEM_C results in its connected locations, such as locations A and B, appearing as the starting and ending locations.

In this procedure, log in to Oracle Forms with Oracle Inventory responsibility and navigate to the Master Item window.

Steps

1. In the Master Item window, query the item, or define it if it does not yet exist.
2. Select the Service tab.
3. In the Install Base section, choose Link from the Instance Class list.
4. Save your work.

Note: You can select the Instance Class only if you have enabled Installed Base Tracking for this item. For more information, see section, "Specify Installed Base Tracking", above

7

Set Up Oracle Bills of Material

This chapter covers the following topics:

- Overview of Set Up Oracle Bills of Material Chapter
- Set Up Oracle BOM

Overview of Set Up Oracle Bills of Material Chapter

This chapter describes setup tasks specific to the Oracle Telecommunications Service Ordering (TSO) solution that you must perform in Oracle Bills of Material (BOM) to implement the TSO solution. For more information on setting up Oracle Bills of Material, refer to the *Oracle Bills of Material User's Guide*.

Before You Begin

Before you can set up Oracle Bills of Material, you must set up Oracle Inventory. For more information, see the chapter, Set Up Oracle Inventory and the Item Master, page 6-1.

Set Up Oracle BOM

In the Oracle TSO solution, you use Oracle Bills of Material to define the structure of Container Models and their contents. A Container Model must be a pick-to-order (PTO) model, and it can include standard items, other PTO models, and Option Classes. The Container Model and its contents must meet certain requirements. For a listing of those requirements, see "Container Model Settings and Structure" in the chapter, Set Up Configurator and Customize the Solution, page 10-1.

For more information on setting up Oracle Bills of Material, refer to the *Oracle Bills of Material User's Guide*.

8

Set Up Oracle Installed Base

This chapter covers the following topics:

- Overview of Set Up Oracle Installed Base Chapter
- Enable Network Configurations
- Create Extended Attributes
- Map Extended Attributes to Items
- Set Up SFM Event Manager Queue Service
- Design Tips

Overview of Set Up Oracle Installed Base Chapter

This chapter describes setup tasks that you must perform to Oracle Installed Base that are specific to the Oracle Telecommunications Service Ordering (TSO) solution.

Before You Begin

Before you can set up Oracle Installed Base for use with the Oracle TSO solution, you must set up Oracle Inventory. For more information, see the chapter, Set Up Oracle Inventory and the Item Master, page 6-1.

Setup Checklist

Following is the setup checklist for implementing TSO-specific functionality for Oracle Installed Base.

Oracle Installed Base-TSO Setup Checklist

Setup Step	Required/Optional
Implement Oracle Installed Base	Required
Enable Network Configurations	Required
Create Extended Attributes	Optional
Set Up SFM Event Manager Queue Service	Required
Map Extended Attributes to Items	Optional

For more information on functionality, setting up, or using Oracle Installed Base, refer to the *Oracle Installed Base Implementation Guide* and the *Oracle Installed Base User Guide*.

Enable Network Configurations

Use the steps in this section to enable network configurations.

Prerequisite

Ensure that the profile option, CSI: Configurator Enabled, is set to Network Models Only. This setting allows the transfer of the network configurations to Oracle Installed Base.

In this procedure, log in to Oracle Forms as System Administrator.

Steps

1. In the System Administration window, navigate to Profiles, System, and select Open.
2. In the Profile field, search for the profile option, CSI: Configurator Enabled. The System Profile Values window opens.
3. On the CSI: Configurator Enabled row in the Site column, choose Network Models Only.
4. Save your work.

Create Extended Attributes

Oracle Installed Base supports user-definable extended attributes at the following

levels:

- **Global:** Global-level extended attributes are applicable to all the item instances in Oracle Installed Base.
- **Item Category:** Category-level extended attributes are applicable to the items of the category for which there are defined extended attributes.
- **Inventory Item:** Item-level extended attributes are applicable to all of the instances of the item type for which there is a defined extended attribute.
- **Instance:** Instance-level extended attributes are applicable only to the instance for which there is a defined extended attribute.

Extended attributes are created as Oracle Installed Base (CSI) lookup of type, CSI_EXTEND_ATTRIB_POOL. When the lookup is created, it is associated with an item, item category, or instance using the Installed Base Extended Attribute Template.

After association with a level, extended attributes can then be mapped to Oracle Configurator attributes or Oracle Service Fulfillment Manager attributes. The item attributes (Features) defined in Oracle Configurator Developer are mapped to these Oracle Installed Base attributes. For details, see Map Install Base Extended Attributes to Configuration Attributes, page 10-22.

> **Note:** For extended attributes used in the TSO flow, the Attribute Name must be unique. It is not sufficient for the combination of Attribute Level and Attribute Name to be unique.

Set Up Extended Attribute Pools

If using extended attributes, implementers must also define the extended attributes that the Installed Base item instances use. This setup is performed by modifying the extended attribute lookup of type, CSI_EXTEND_ATTRIB_POOL.

In this procedure, log in to Oracle Forms with Oracle Installed Base Admin responsibility and navigate to Install Base Lookups.

Steps

1. Search Install Base Lookups for the type, CSI_EXTEND_ATTRIB_POOL. A list of available lookups appears.
2. Enter a new lookup Code, Meaning, and Description.

Example:

- Code = PHONE_NUM
  - Meaning = Phone Number

- Description = Phone Number

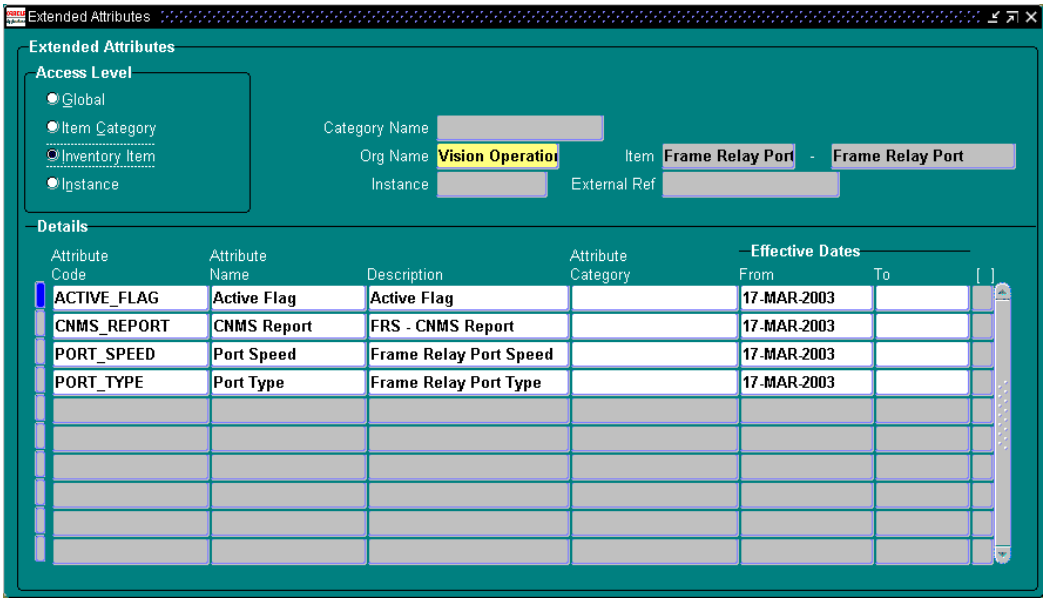
- Save the new lookup.

Map Extended Attributes to Items

In this procedure, log in to Oracle Forms with Oracle Installed Base Admin responsibility and navigate to Setups, Extended Attribute Template. Map extended attributes or other attributes (lookups) available in the CSI_EXTEND_ATTRIB_POOL to the Service Fulfillment Items. As a prerequisite, you must have defined the item and extended attributes.

The following figure shows the extended attribute and its details of an inventory item.

Extended Attribute and Details of an Inventory Item



Set Up SFM Event Manager Queue Service

Ensure that Oracle Service Fulfillment Manager Event Manager Queue Service is set up correctly. Use the following steps:

Steps

1. As SFM System Administrator, navigate to Concurrent, Define Manager.

2. Query for SFM Event Manager Queue Service and select the Work Shifts Button. For all defined workshifts, check the following:

- The value for the Processes is equal to 1
- In the parameters field, find the value, XDP_DQ_INIT_THREADS. Ensure that XDP_DQ_INIT_THREADS is equal to 1.

  If either of the above steps do not work, bounce the Event Manager Service (see "Start the Queue"/"Stop the Queue" steps, below).

3. Start the queue: Navigate to Administrator, Queue Console.
4. Select the Services button and then select the record for SFM Event Manager Queue Service.
5. Select Restart.

   If the above action does not start the queue, follow the below steps:

Stop the Queue

1. Using SFM System Administration responsibility, navigate to Administrator, Application.
2. Stop it is a SRS / Conc. program. Parameters: ALL, null, NORMAL.

Start the Queue

1. Using SFM System Administration responsibility, navigate to Administrator, Application.
2. Start (it is a SRS / Conc. program. Parameters: ALL, null, 'anything' – it is a debug level, your need for details should drive it, default 0)

Note: If this setup is incorrect, the transactions are forcibly errored out into the Oracle Installed Base error processing schema. Given below are the steps to verify if the setup is correct. These steps can verify the setup in any case where the Sales Order lines are FULFILLED but the expected IB updates did not happen and the SFM Event Manager Queue Service is functional and found to be enqueuing/dequeuing the messages.

1. As Install Base Administrator, navigate to Transaction Errors Processing.
2. Provide the query criteria; preferably this would be dates to catch all types of exceptions.

Design Tips

Implementers must manually verify that the attributes set up in Oracle Configurator Developer are coordinated with the Oracle Installed Base extended attributes setup for trackable items.

Configured attributes which are modeled as Installed Base extended attributes must

only be set up with one level. Oracle Configurator does not model the attribute level as part of the configured attribute definition. Therefore, the TSO solution does not support the use of extended attributes with multiple-level definitions.

See also: CZ_CONFIG_EXT_ATTRIBUTES table in the chapter, Set Up Configurator and Customize the Solution, page 10-1, and the section, "IBAttribute Configurator Extension", in the chapter, Set Up Configurator Extensions, page 11-1.

9

Set Up Oracle Advanced Pricing

This chapter covers the following topics:

- Overview of Set Up Oracle Advanced Pricing Chapter
- Set Up Recurring and One-Time Charges
- Advisory: Order Amount and Cross-Order Volume Computations
- Set Up Pricing Attributes and Sourcing Rules

Overview of Set Up Oracle Advanced Pricing Chapter

This chapter describes setup tasks that you must perform to Oracle Advanced Pricing that are specific to the Oracle Telecommunications Service Ordering (TSO) solution.

Before You Begin

Before you can set up Oracle Advanced Pricing for use with the Oracle TSO solution, you must set up Oracle Order Management. For more information, see the chapter, Set Up Oracle Order Management, page 5-1.

> **Important:** The Order Management profile, OM: UOM Class for Charge Periodicity, must be set prior to creating price list or modifier list setups for recurring charges in Advanced Pricing.

Setup Checklist

Following is the setup checklist for implementing TSO-specific functionality for Oracle Advanced Pricing.

Oracle Advanced Pricing-TSO Setup Checklist

Setup Step	Required/Optional
Implement Oracle Advanced Pricing	Required
Set Up Recurring and One-Time Charges	Required
Set Up Pricing Attributes and Sourcing Rules	Optional

For more information on functionality, setting up, or using Oracle Advanced Pricing, refer to:

- *Oracle Advanced Pricing Implementation Manual*
- *Oracle Advanced Pricing User's Guide*

Set Up Recurring and One-Time Charges

The functionality in Oracle Advanced Pricing is available for the products with recurring charges as well as the products with one-time charges.

Charge Periodicity Attribute and Recurring Charges

Oracle Advanced Pricing seeds a pricing attribute, Charge Periodicity, to pass charge periodicity from the ordering application to Oracle Advanced Pricing. Implementers leverage charge periodicity by attaching the Charge Periodicity pricing attribute to a price list line or a modifier line. Some services, such as wireless phone services, may charge their customers a fixed recurring service charge for certain services: for example, a monthly charge for a local or long distance calling plan, or other services, such as caller ID or call waiting. The billing frequency of these charges, such as every month or quarter, is the charge periodicity. Service providers may choose to offer customers the same service in different bundles (plans) with each plan having different charge periodicities. For example, a wireless phone company can offer the same phone service with either a monthly price or a quarterly price. These recurring charges are typically set up when the customer signs up for the service.

When a customer orders an item with a recurring charge, the periodicity must be specified on the order, quote, or shopping cart line to get the correct price for the service. To provide for periodicity entries, an optional field called Charge Periodicity is available to the order and quote line of hosting applications to hold the periodicity. When setting up a recurring charge periodicity pricing attribute for a price list line or modifier line, implementers can select from seeded (in Oracle Inventory) unit-of-measure (UOM) periodicity codes such as month, quarter, and year; alternately,

implementers could define a UOM Class with customized UOM codes. The following seeded values can be selected when setting up charge periodicities for either a price list line or modifier line:

- Pricing Context: Periodicity
- Pricing Attribute: Charge Periodicity
- Values: Units of measure associated to UOM class specified in the profile, OM: UOM Class for Charge Periodicity (e.g., Month, Quarter, Year)

Advisory: Order Amount and Cross-Order Volume Computations

Computations for Order Amount qualifier and Cross-Order Volume attributes only include one-time charge lines; recurring charge lines are excluded from these calculations.

Set Up Pricing Attributes and Sourcing Rules

Pricing objects in Oracle Advanced Pricing enable users to define pricing actions and pricing rules for a given business process. Through price lists, modifiers, and formulas, these pricing actions provide the ability to define prices, price adjustments, and other benefits. By defining pricing rules, such as qualifiers and pricing attributes, implementers use these pricing rules to drive pricing actions.

Through attribute management, the pricing engine receives all of the values of the attributes defined in the qualifier and pricing attributes to determine which price lists and modifiers the transaction is qualified for. The calling application first uses the attribute mapping API to get all qualifier and pricing attributes associated with the transaction. Then, the calling application makes a call to the pricing engine. The pricing engine evaluates these values to determine which price lists and modifiers are eligible for the transaction.

The attribute management feature lets implementers create new contexts and attributes, update existing context or attribute properties, or disable existing contexts. Creating new context or attributes extends the Oracle Advanced Pricing ability to provide user-defined data sources that drive pricing actions. The methods to source data for an attribute are:

- **User-entered:** User enters attribute value
- **Custom-sourced:** Uses custom code to derive an attribute
- **Attribute mapping:** Attribute values are derived from Oracle or non-Oracle data sources based on sourcing rules that are executed within the Build Sourcing API called before the pricing engine call

- **Runtime-sourced:** Attribute value is sourced within the pricing engine

For example, in the context of TSO, attribute management can be used to derive the price of a Frame Relay PVC depending upon the PVC Speed, which could be both an Oracle Installed Base attribute and an Oracle Configurator attribute. You can custom-source the value of this attribute and provide it to the pricing engine to derive the price based on this attribute value.

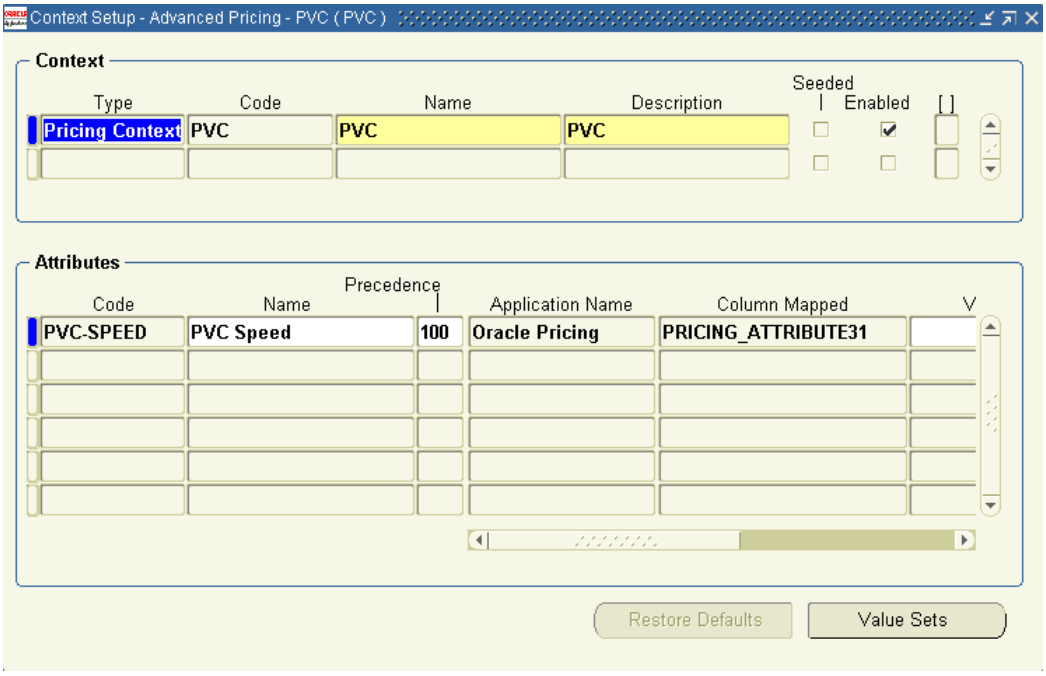
The following sections provide the setup steps required in Advanced Pricing to implement this example scenario.

Attribute-Based Pricing Setup

Attribute-based pricing setups are done in the Context Setup form of Oracle Advanced Pricing. The navigation path is: Pricing, Setup, Attribute Management, Context and Attributes.

The following figure shows an example of the Context Setup form, with the setting up pricing attributes for PVC Speed.

Context and Attributes Sample Setup

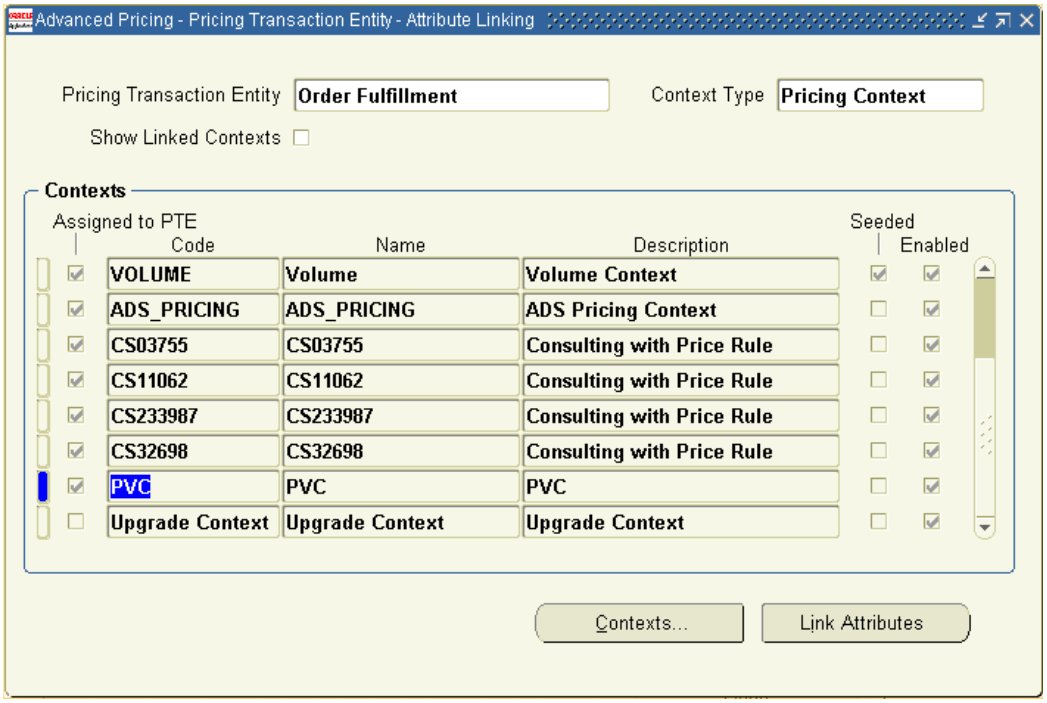


Attribute Linking and Mapping

Attribute linking and mapping is done in the Attribute Linking form of Oracle Advanced Pricing. The navigation to Attribute Linking and Mapping setup is: Pricing, Setup, Attribute Management, Attribute Linking and Mapping.

The following figure shows an example of pricing attribute linking and mapping for PVC in the Attribute Linking form.

Attribute Linking and Mapping Sample Setup

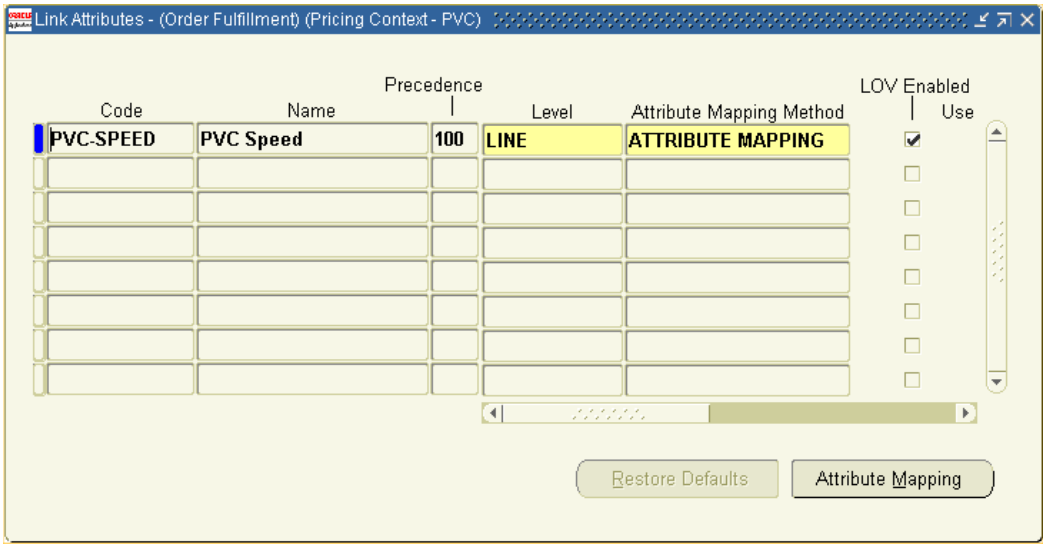


Attribute Linking

The navigation to Link Attributes setup is: Pricing, Setup, Attribute Management, Attribute Linking and Mapping, Link Attributes

The following figure shows an example of the attribute linking.

Attribute Linking Sample Setup

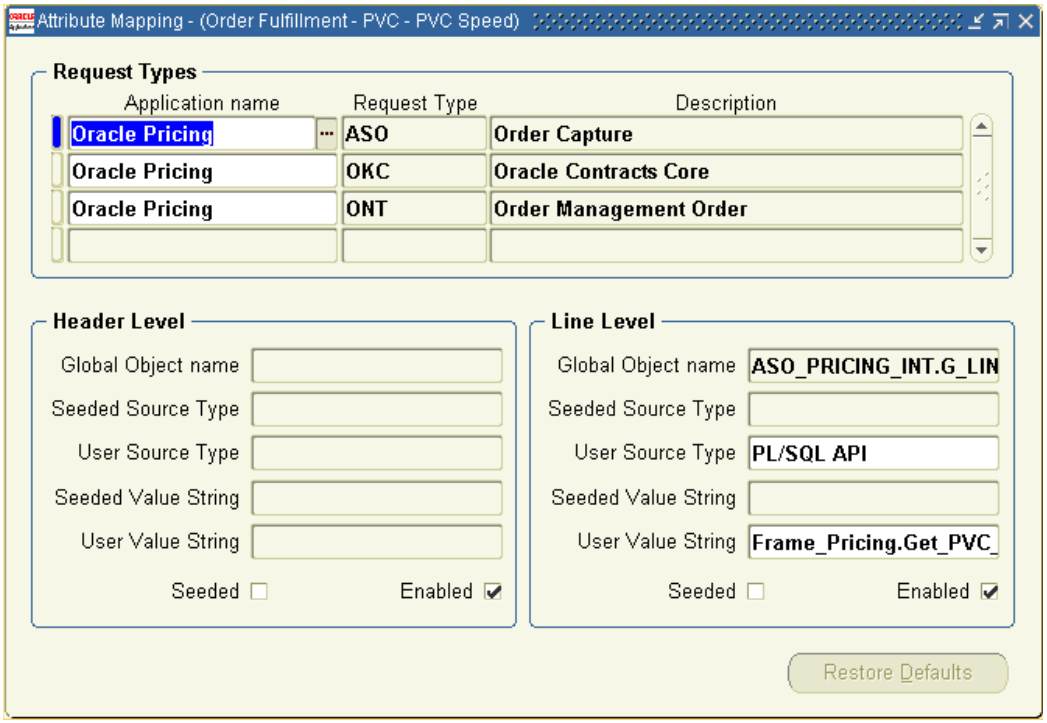


Attributes Mapping

The navigation to Attributes Mapping is: Pricing, Setup, Attribute Management, Attribute Linking and Mapping, Link Attributes, Attributes Mapping.

The following figure shows an example of attribute mapping.

Attribute Mapping Sample Setup



The global parameter quote_line_id can be used to link by way of the configurator keys to the CZ_Config_Ext_Attributes table to select the attribute value for by the attribute name. The value returned is sourced to the Pricing Attribute PVC_Speed. This pricing attribute can either be used in a Formula and associated to a price list or can be just used to define various prices for an item.

10

Set Up Oracle Configurator and Customize the Solution

This chapter covers the following topics:

- Overview of Set Up Oracle Configurator Chapter
- Configurator Concepts
- Configurator Implementation in the Solution
- Disable Pricing
- Map Install Base Extended Attributes to Configuration Attributes
- Oracle Configurator Schema Customizations
- Initialization Parameters
- Batch Validation Parameters
- Programmatic Tools for TSO Instance Management

Overview of Set Up Oracle Configurator Chapter

This chapter describes setup tasks that you perform to Oracle Configurator that are specific to the Oracle Telecommunications Service Ordering (TSO) solution.

Before You Begin

Before you can set up Oracle Configurator for use with the Oracle TSO solution, you must set up Oracle Bills of Material, Oracle Installed Base, and Oracle Order Management.

In addition to this chapter, there is another chapter on setting up Oracle Configurator for use with the TSO solution, Set Up Oracle Configurator Extensions, page 11-1.

Setup Checklist

Following is the setup checklist for implementing TSO-specific functionality for Oracle Configurator.

Oracle Configurator-TSO Setup Checklist

Setup Step	Required/Optional
Implement Oracle Configurator	Required
Create Configuration Model	Required
Publish Container Model	Required
Define Profile Options	Required
Disable Pricing	Optional
Customize Oracle Configurator	Optional

For more information on functionality, setting up, or using Oracle Configurator, refer to:

- *Oracle Configurator Developer User's Guide*
- *Oracle Configurator Implementation Guide*
- *Oracle Configurator Installation Guide*
- *Oracle Configurator Extensions and Interface Object Developer's Guide*
- *Oracle Configurator Methodologies*
- *Oracle Configurator About documentation*

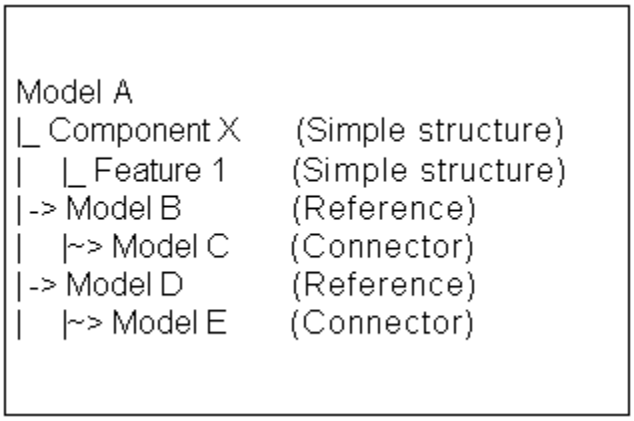
Configurator Concepts

This section discusses Oracle Configurator concepts you need to be familiar with before implementing a TSO solution.

Conventions for Connector and Components of a Model

Some of the diagrams that appear in this chapter show the structural relationship

between Connectors and Components of a Model. The following sample diagram and table show and describe the conventions for the Model diagrams.



The following table refers to the preceding diagram.

Model Diagram Conventions

Diagram Part	Description
Model A |_ Component X	This part represents a simple Model structure. In this case, Component X is a child of Model A.
| |_Feature 1	Feature 1 is a child of Component X.
|->Model B	This represents a Reference node. Model A references Model B.
| |~> Model C	This represents a Connector node. Model B has a Connector to Model C.

Using Container Models

You can mark Oracle Bills of Material (BOM) Models within a Container Model as trackable. You can also mark BOM Option Classes and BOM Standard Items as trackable if they are inside a BOM Model. Tracking enables Oracle Installed Base to record information about an instance of that item, including its location, status, current configuration, change history, and so on. You must define the Container Model itself as non-trackable. Oracle Installed Base does not record information about non-trackable items. For more information about making an item Installed Base trackable, see the chapter, Set Up Installed Base.

Before an end user can reconfigure any of a Container Model's installed configurable instances, the Container Model must meet certain requirements, such as a valid Container Model structure, Connectors, and rules, to ensure that the configured items are trackable in Oracle Install Base. For more information about the necessary requirements, see the section, "Container Model Settings and Structure", below, and the chapter, Set Up Install Base.

Container Model Settings and Structure

The following list identifies the requirements for a Container Model in Oracle Inventory and Oracle Bills of Material.

A Container Model*must* be:

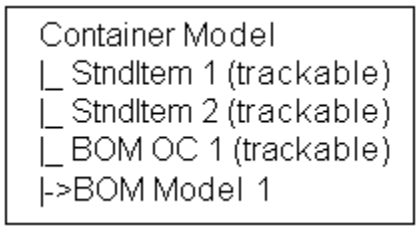
- A Pick-to-Order (PTO) BOM Model

A Container Model *cannot*:

- Be trackable
- Reference another Container Model. In other words, a Container Model cannot be a Component within another Container Model
- Contain more than one effective reference to the same trackable BOM Model; for more information about references and effectivities, see the *Oracle Configurator Developer User's Guide*.
- Contain a non-trackable BOM Model with trackable children--one or more BOM Option Classes or BOM Standard Items
- Contain any BOM Models or BOM Option Classes that have a default quantity greater than 1.
- Have any trackable BOM Standard Items or trackable BOM Option Classes as direct children

For example, the following figure shows BOM trackable items StndItem1, StndItem2, and BOM OC 1. This structure is invalid because these BOM items are all direct children of the Container Model.

Invalid Container Model Structure



Container models can contain Standard Items that are tangible. See Tangible Items, page 10-14 for details.

For more information on creating or specifying Container Models, see the chapter, Set Up Inventory.

Importing Container Models into Oracle Configurator

To develop a Container Model in Oracle Configurator Developer, you must import the Oracle Inventory and BOM data into the Oracle Configurator (CZ) schema of the Oracle Applications database. For information about populating the CZ schema, see the *Oracle Configurator Implementation Guide*.

The concurrent programs under Populate and Refresh Configuration Models in Oracle Applications import the completed bills of material into the Oracle Configurator schema. Run the concurrent programs from the Oracle Configurator Administrator menu.

These concurrent programs import BOM structure (assemble-to-order, PTO, or Container Models, structure, and rules) and require complete BOMs that are identified at the desired root.

The concurrent programs under Populate and Refresh Configuration Models in Oracle Applications include:

- **Populate Configuration Models**: Populates the CZ schema online tables with data that is appropriate for creating configuration Models that are based on existing BOM or legacy data.

- **Refresh a Single Configuration Model**: Updates a specific BOM Model with any changes that may have been made in Oracle Applications since the time the BOM was first imported into the CZ schema.

- **Refresh all Imported Configuration Models**: Updates all of the BOM Models that were imported into the specific database instance that the user is logged into.

- **Disable/Enable Refresh of a Configuration Model**: Enables the user to prevent specific configuration Models from being refreshed if the Refresh all Imported Configuration Models concurrent program is run.

When importing a container BOM Model, the system creates a Model node in Oracle Configurator Developer. If the imported Model has Components that are assemble-to-order (ATO) or PTO Models, then the import process creates:

- A Model structure for each child Model.
- A corresponding reference node in the parent Model.

If a Standard Item is marked as Shippable, Inventory Transactable, and Serializable, then it is imported as a tangible item. See Tangible Items, page 10-14 for details.

Note that the importing of a BOM Model into Oracle Configurator Developer is not TSO-specific functionality. The preceding section sets the context for what may happen when a BOM Model that is also a container is imported or refreshed into Oracle Configurator Developer.

Verification Abilities of Populate and Refresh Concurrent Programs

The concurrent programs under Populate and Refresh Configuration Models do *not* verify that all of the settings that you selected when defining a Container Model in Oracle Applications are correct. If a Container Model violates any of the requirements listed in this section, an error occurs only when you generate logic in Configurator Developer, not when you import or refresh the Model.

The Refresh a Configuration Model and the Refresh all Imported Configuration Models concurrent programs check only two settings that you specify in Oracle Applications. When you *refresh* a Container Model, the concurrent program displays:

- A warning, if the Configuration Model Type changed from Container to Standard. The Refresh concurrent programs do not display a warning if you changed the Configuration Model Type from Standard to Container.
- An error, if any of the Container Model's child BOM Models capable of multiple instances changed from a PTO to an ATO BOM model. For example, the Instantiability Initial Minimum and Initial Maximum values for Model X (a PTO Model) are not equal in Configurator Developer. In Oracle Inventory, you change Model X from a PTO Model to an ATO Model. When you refresh the Container Model, the concurrent program displays an error.

Structuring Container Models

In a Container Model, if one of the trackable child Models does not have a trackable parent or ancestor, there cannot be a limit on how many instances of that trackable child Model can be created at runtime. In other words, the trackable BOM Model's Instantiability setting must be Multiple or Variable Instances with an Initial Minimum of 0 and Initial Maximum of null in Configurator Developer.

If the Initial Minimum and Initial Maximum values are not 0/null, respectively, for a trackable item, Configurator Developer displays an error when you generate logic. A

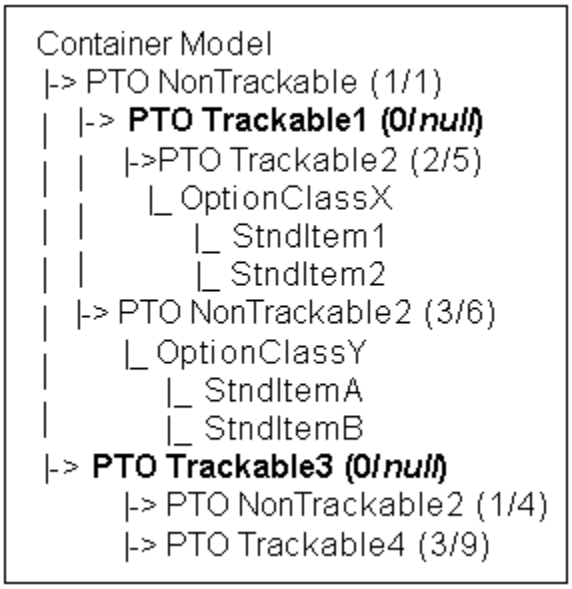
trackable item can contain both trackable and non-trackable items, such as BOM Models and Option Classes, as well as Components created in Configurator Developer.

Note: The first time you import a Container Model, the Populate Configuration Models concurrent program sets the Initial Minimum and Initial Maximum values to 0/null, respectively, for all trackable child BOM Models. However, *refreshing* a Container Model does not reset these values if you change them in Oracle Configurator Developer. If the Model's structure violates any of the requirements listed in this section, Oracle Configurator Developer displays an error when you generate logic.

Because you can create multiple instances of such a BOM Model during configuration, and Oracle Install Base tracks these instances when they are installed, a BOM Model matching this criteria is a trackable item. An example of a trackable item appears in the figure below.

Remember that you can only track BOM items, and that Oracle Install Base only records information about trackable BOM items.

Example of Trackable Items



In the above figure, PTO Trackable1 and PTO Trackable3 are trackable items because they are themselves trackable and do not have a trackable parent or ancestor. Therefore, you must set the Instantiability Initial Minimum and Initial Maximum values to 0 and null, respectively. The figure shows these settings as 0/null.

Note that the Initial Minimum and Initial Maximum values for PTO Trackable 2 and PTO Trackable 4 are *not* set to 0/null. This is because a trackable Model that is a *child* or a *descendant* of a trackable item does not have the same configuration requirements as a trackable item. That is, the trackable child item does not have to obey the rule when no instances exist, as long as the following is true:

- The configuration session begins
- There is no limit to how many instances the end user can create

Refer to the *Oracle Configurator Developer User's Guide* for more information about creating multiple instances of Components during configuration.

Examples of Valid and Invalid Container Model Structure

This section provides examples of how trackable items can impact the validity of a Container Model's structure in Oracle Configurator Developer.

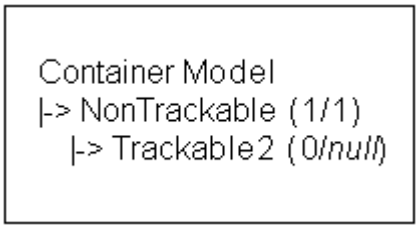
In a Container Model, a trackable descendant Model can be a child of a non-trackable Model only if the non-trackable Model's Initial Minimum and Initial Maximum values are both set to 1, and:

- The trackable Model's Initial Minimum and Initial Maximum values are 0/null (see the figure, Valid Model Structure: Non-Trackable Parent and Trackable Child, below)

or

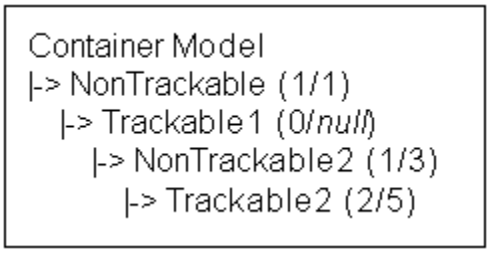
- Another trackable Model whose instances values are 0/null is the parent of the non-trackable Model (see the figure, Valid Model Structure: Trackable Child and Trackable Ancestor, below)

Valid Model Structure: Non-Trackable Parent and Trackable Child



In the figure above, the Model structure is valid because the Initial Minimum and Initial Maximum for Trackable2 equals 0/null.

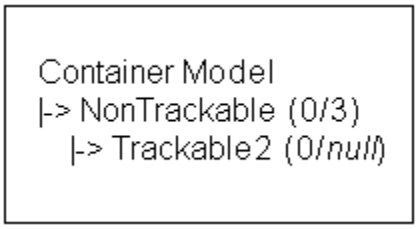
Valid Model Structure: Trackable Child and Trackable Ancestor



Although Trackable2 is a child of NonTrackable2, the structure shown in the second figure is valid because the Initial Minimum and Initial Maximum for Trackable1 equal 0/null and Trackable1 is an ancestor of Trackable2. Because Trackable1 is a trackable item, an Oracle Install Base or Oracle Quoting user can select an installed instance of Trackable1 and launch Oracle Configurator to reconfigure it.

The following shows an invalid structure. It is invalid because NonTrackable's Initial Minimum equals 0.

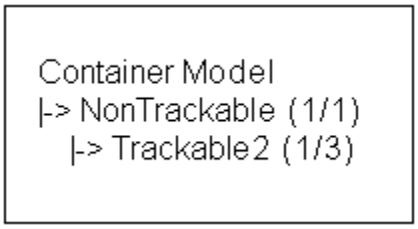
Invalid Model Structure: Non-Trackable Parent and Trackable Child



If an Oracle Install Base or Oracle Quoting user selects an instance of Trackable2 for reconfiguration, its parent (NonTrackable) is not part of the configuration when the Oracle Configurator session begins. Therefore, Oracle Configurator cannot add an instance of Trackable2 to the configuration either.

In the following figure, Trackable2 is the first trackable BOM Model in the Container Model's structure. However, its instances values are not 0/null.

Invalid Model Structure: Non-Trackable Parent and Trackable Child



This structure is invalid because although you can configure and track Trackable2 in Oracle Install Base, there is a restriction on how many instances of Trackable2 are allowed at runtime.

Connecting Components

Oracle Install Base tracks the relationships that connections form between Components. Oracle Install Base keeps a record only of *trackable* BOM items, so any connections made at runtime must be from an instance of a trackable BOM Model to instances of other trackable BOM Models.

Because Oracle Install Base keeps a record of connected instances only if they are

trackable, any Connectors that you create within a Container Model in Oracle Configurator Developer must be valid. If a Container Model contains Connectors that violate any of the requirements listed in this section, Oracle Configurator Developer displays an error when you generate logic.

Connectors enable you to define rules that include nodes from trackable items as participants. For more information, see the section, "Using Configuration Rules", above.

When working in a Container Model in Oracle Configurator Developer, you must create Connectors only from:

- A trackable BOM Model to another trackable BOM Model
- A non-trackable BOM Model to another non-trackable BOM Model

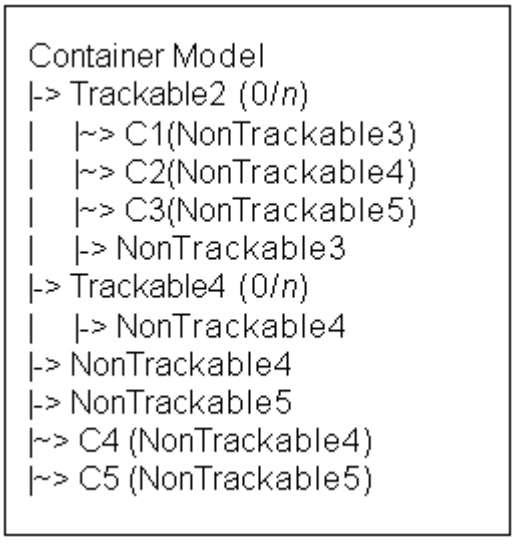
You must *not* create Connectors:

- From the Container Model's root node to a trackable Model
- From the Container Model's root node to any Model that is a child of a trackable Model

For example, a Container Model references a trackable Model named PTO1. PTO1 references a non-trackable Model named PTO2. You create a Connector from the Container Model to PTO2. Because you created a Connector from the Container Model to a Model that is a child of a trackable Model, Oracle Configurator Developer displays an error when you generate logic.

- Within a trackable Model to a non-trackable Model that has the trackable Model as an ancestor (see the following figure)

Invalid Connectors



In the above figure:

- Connectors C1, C2, and C3 are invalid because you cannot connect a trackable Model and a non-trackable Model.
- Connectors C4 and C5 are allowed because the Connectors' parent and target are both non-trackable Models (in other words, they do not cause an error when you generate logic)

But there are differences. Connector C5 has only a non-trackable target in NonTrackable5. Connector C4 has a non-trackable target with Model NonTrackable4 being a child of the Container. The Model NonTrackable4 is also a child of Trackable4, but these instances would be excluded from consideration by the Connector, and will not appear in the list of eligible targets.

Note: In Oracle Configurator Developer, when you select a BOM Model node, the BOM Item Type field indicates whether it is an ATO, PTO or Container Model. Additionally, when you select any BOM node, the Trackable check box indicates whether the item is trackable. You can expand the Definition attribute to view these settings.

Network Link Items

A network Link item is a BOM Model that connects trackable items. At runtime, an instance of a Link item connects instances of trackable items. In the TSO context, a Link item has only one unique aspect: it does not require that a location be assigned to it. For an explanation of the need for location information, see Locations, page 11-6.

An item is defined as a Link in Oracle Inventory, as described in Set Up Link Items, page 6-6. To determine at runtime whether a trackable item instance is a Link, Oracle Configurator checks whether the item's link flag is set to true. The link flag is described in Link Items Column, page 10-28. The link flag setting for a BOM Model is part of the structure that is imported with the Model.

In Oracle Configurator Developer, you must complete the definition of a Link by adding exactly two child Connectors to a BOM Model having a link flag set to true, to specify the connections that are valid between trackable items at runtime.

Transient Items and Attributes

A transient item is a BOM Standard Item used for the purpose of pricing a non-recurring service or fee. This concept is narrower than that of a one-time charge.

The ordering of a wireless phone may be a one-time charge that requires a 1-time fulfillment. This charge can also be trackable, in which case it reappears during a subsequent reconfiguration of the service. Another type of charge associated with changing a service, such as an installation fee or a service termination fee, is also a 1-time charge, but should never reappear during a subsequent reconfiguration of a service. Such items are termed *transient*.

Transient items are *not* trackable. They are never installed as instances in Installed Base.

They do not appear in Oracle Configurator when the item they were initially ordered with is reconfigured. To ensure that these items do not appear during reconfiguration (where they might confuse the end user), you must use the *transient flag* to designate the transient items in your configuration Model.

Transient items can never be discontinued. Consequently, the method `RuntimeNode.isDiscontinued()` always returns false when used on a node that is a transient item or attribute. See About Discontinued Items, page 11-33 for an explanation of discontinuation.

Since only BOM Standard Items can be transient, and since BOM Standard Items are always leaf nodes (i.e., nodes without children) in a configuration Model hierarchy, transient items are always leaf nodes.

Flagging Transient Items

You flag items as transient in Oracle Configurator Developer.

If a node satisfies all of the following criteria, then the Transient checkbox is enabled:

- Is a BOM Standard Item (not a BOM Model or BOM Option Class)
- Being a BOM Standard Item requires that the node is a leaf node (has no child nodes)
- Is *not* trackable (the Trackable flag is not set)

To mark an item as transient:

1. In Main area of the Repository, edit the BOM Model containing the BOM Standard Item.
2. Navigate to the Structure area of the Workbench.
3. Edit the BOM Standard Item.
4. In the Details page for the BOM Standard Item, select the Transient checkbox.
5. Click Apply.

See the section, "Transient Items Column", below for related database details.

Transient Attributes

A transient attribute is conceptually similar to a transient item. After a service is fulfilled, attributes that are transient are not restored during the reconfiguration of the service.

You model transient attributes in Configurator Developer, with Features. Any type of Feature (Integer, Decimal, Boolean, Text, or Option Feature) can be used as a transient attribute.

During the reconfiguration of a trackable instance, Oracle Configurator ignores any inputs that are values of features flagged as Transient.

Attributes are assigned by implementing the IBAttribute Configurator Extension. See the chapter, Set Up Configurator Extensions, for details.

Flagging Transient Attributes

You flag attributes as transient in Oracle Configurator Developer. If node satisfies all of the following criteria, then the **Transient** checkbox is enabled.

- Is a Feature (Integer, Decimal, Boolean, Text or Option Feature)
- Is not an Option of an Option Feature
- Is not a Resource or a Total

Do not choose an Option Feature with a Maximum greater than 1. This will cause an error at runtime.

To mark an item as transient:

1. In Main area of the Repository, edit the Model containing the Feature.
2. Navigate to the Structure area of the Workbench.
3. Edit the Feature.
4. In the Details page for the Feature, select the Transient checkbox.
5. Click Apply.

Assumptions and Restrictions

During the design and testing phases of your solution, you must ensure that the following issues do not cause problems.

- When a saved configuration of a trackable instance is restored for reconfiguration, the Oracle Configuration Interface Object (CIO) ignores all configuration inputs that have been flagged as transient, and does not report any deltas for the transient entity.
- The removal of transient items or attributes at reconfiguration can result in changes to the set of selected items. In addition, if you have defined constraint rules between transient and non-transient entities, then logical contradictions may result when the transient participants to the rules are not restored.
- Assertions to transient items or attributes must be performed by end user selections or by Configurator Extensions. Transient items or attributes should not participate in constraint rules. (Assertions are changes to the state, value, or quantity of an

entity.)

Tangible Items

Physical BOM standard items (or Kits) in a container model that are shippable, inventory transactable, and serializable are called tangible items.

See "Specify Install Base Tracking" in the chapter "Set Up Oracle Inventory" for details on how to make an item tangible.

This functionality enables a sales representative to provide customers with the option to purchase shippable items (such as wireless handsets or Customer Premises Equipment (CPE)) while ordering services.

Oracle Configurator allows the end user to perform these actions related to tangible items:

- **Create new configurations with tangible items in a Container model.**

- **Return tangible items while there are pending orders.**

- **Reconfigure trackable component instances that contain returned tangible items.**

  When restoring a configuration that has items that are flagged as returned, the CIO restores the "returned" state on the corresponding runtime node. This state is then used to remove the tangible item from the configuration. The removal is not permanent until the reconfiguration order is booked and fulfilled.

  If the model's configuration rules require the existence of a tangible item, but that item has previously been returned (removed from the configuration), then a failure message informs the user that this item was previously returned but is still required by the system and that the item is treated as a newly repurchased item in the current (reconfiguration) session. If the user does not want this item in the configuration, then he or she must explicitly remove it. To avoid this scenario, avoid defining rules that require the existence of tangible items, or define rules in such a way that they can be relaxed during the reconfiguration flow through Configurator Extensions.

- Note that associating extended attributes to tangible items is *not* supported.

In order to be a tangible item in a Container Model, an item must meet the following restrictions:

- Must be a non-ATO standard item, or a kit

- Must be flagged as Shippable, Inventory Transactable, and Serializable

  Tangible items must be flagged as Serializable in *all* inventory organizations. Otherwise, there may be errors in the configuration flow that introduce corruption

into Install Base. Since the Inventory Transactable flag can only be set in the master organization, this restriction does not apply to it.

- Does not have any extended attributes on it
- Has a maximum quantity of 1
- Does not have any Connectors to it

If a tangible item is discontinued then only the transaction relationship (which will have relationship type "Component-Of") corresponding to that item is discontinued. The discontinued shippable item remains in Oracle Install Base.

You can only add or delete tangible items from a configuration. You cannot change the state of tangible items relative to their state in Oracle Install Base.

When defining configuration Rules in Oracle Configurator Developer, Since a tangible item cannot have a quantity greater than 1, you cannot define a numeric rule in Oracle Configurator Developer that contributes to or consumes from the quantity of a tangible item.

Instance Locking for Trackable Items

It is possible for Oracle Install Base data to become corrupted if orders become stacked. In previous releases, a stacked order could occur when a trackable instance of an item was reconfigured before the fulfillment of the previous revision of the instance was complete. This situation could arise when the previous fulfillment failed or was still pending when the item was reconfigured and booked for fulfillment.

To prevent data corruption, Oracle Configurator locks instances until they are completely fulfilled, to ensure the integrity of their revision data. Specifically:

- When an order is booked, the instance being reconfigured is locked against further reconfiguration until fulfillment is completed.
- When a booked order is cancelled or when it is fulfilled and processed through to Install Base, the locked instance is unlocked.
- If the instance has been locked by a previous order, the new order cannot be booked, and the reconfiguration is invalidated. The end user is informed of the invalidation by a message.
- A locked instance cannot be updated directly in Install Base.
- Item instances connected to locked instances are also locked.
- The CIO provides the method `ITrackableInstance.isLocked()`, which returns true if an instance is locked.

Using Configuration Rules

Oracle Quoting and Oracle Install Base end users typically need to reconfigure only a few Components within a network of *connected* Components. To facilitate this process and improve runtime performance, you can view only the following Components when an Oracle Configurator session begins:

- The Components selected for reconfiguration
- Any Components required to enforce constraints defined in the Model

When an end user reconfigures an installed configuration, Oracle Configurator may need to add Components to the configuration session to ensure the validity of all connections and the entire configuration. At runtime, only rules that you define using *Connectors* can propagate to Component instances that may not be visible in the configuration session. For this reason, it is important to define rules using Connectors when any rule participants are not part of the same trackable item. In other words, no parent-and-child or ancestor-and-descendant relationship exists between the trackable items. For more information on trackable items, see the section, "Structuring Container Models", above.

Note: If two trackable items are part of the *same* trackable item, it is not necessary to create a Connector between them in order to define a rule in which they are both participants. For example, Model A is a trackable BOM Model that has two children, Model B and Model C. Since Model A *references* these Models (thus they are Model A's children), you can use nodes from both Model B and Model C when defining rules.

When a rule defined using Connectors propagates to instances that are not visible in the configuration session, Oracle Configurator prompts you to add instances to the configuration session so other required changes are applied and the configuration remains valid.

For more information on using Connectors to define rules, see the *Oracle Configurator Developer User's Guide*.

Valid Configuration Rule Combinations

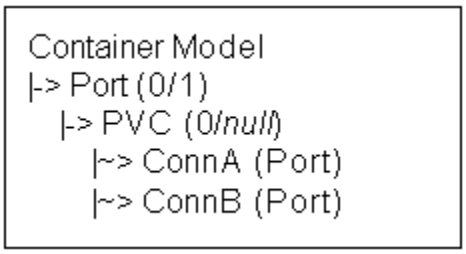
The following are examples of valid configuration rules for trackable Models within a Container Model.

- A rule can contain participants from a trackable Model and one of its trackable child Models.
- You can create a rule between Connectors that have the same trackable Model as their target.
- A rule that includes nodes from two non-trackable Models is valid as long as *neither* Model is a child of a trackable Model, or they are *both* children of the *same* trackable

Model.

Using the structure in the following figure as an example, you can create rules using nodes from Connector A and Connector B, even though their targets are the same. (You can also create rules using nodes from Connector A and Connector B if their targets are different.)

Connectors to the Same Trackable Model



Invalid Configuration Rule Combinations

If a configuration rule violates any of the requirements in this section, Oracle Configurator Developer displays an error when you generate logic.

The following are examples of *invalid* configuration rules for trackable Models within a Container Model.

- A numeric rule that dynamically changes how many allowable instances of a *trackable* child BOM Model can exist at runtime. This type of rule changes how many instances of a Model are allowed at runtime.

- A numeric rule in which a Container Model node contributes to or consumes from the minimum or maximum number of allowed instances at runtime for another Model (regardless of the Model's type). For more information about this type of numeric rule, see the *Oracle Configurator Developer User's Guide*.

- A rule that uses nodes from sibling trackable Models (that is, the Models exist at the same level in the Container Model's structure).

- A rule between a trackable Model and a non-trackable Model that is not one of the trackable Model's children (see example below).

- A rule with participants from one trackable Model and any other node from a non-trackable child of the Container Model.

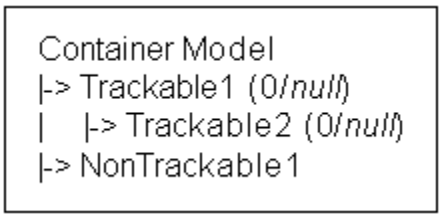
- A numeric rule with a tangible BOM Standard Item that is as a participant in Operand 2. You cannot define a numeric rule that contributes to or consumes from the quantity of a tangible item, because a tangible item cannot have a quantity greater than 1.

Invalid Container Model Structure Examples

Example 1: A rule between a trackable Model and a non-trackable Model that is not one of the trackable Model's children.

Using the structure shown in the following figure, create a rule containing participants from Trackable1 and NonTrackable1. This rule is invalid and causes an error when you generate logic. This restriction does not apply to Configurator Extensions.

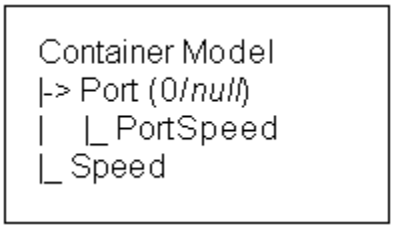
Container Model Structure



Example 2: A rule with participants from one trackable Model and any other node from a non-trackable child of the Container Model.

For example, in the structure appearing in the following figure, both PortSpeed and Speed are Features. A rule using PortSpeed and Speed is invalid because Port is trackable, PortSpeed is a Feature in Port, and Speed is a direct child of the Container Model.

Container Model Structure

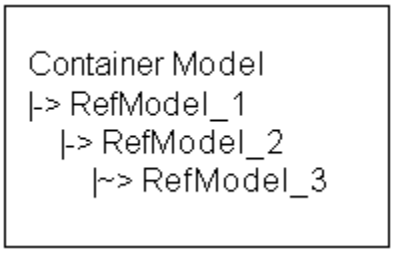


Example 3: An invalid rule in a referenced Model.

In the structure appearing in the following figure, the Container Model contains a tangible item, and the referenced model RefModel_3 contains a numeric rule that contributes a quantity greater than 1 to the tangible item. The numeric rule is invalid, because a tangible item cannot have a quantity greater than 1. This invalid rule causes an error when you generate logic in RefModel_3, which contains the rule. However, this invalid rule does *not* cause an error when you generate logic in Container_Model, if the logic in the referenced model RefModel_3 is already up-to-date. You can force Oracle Configurator to generate and validate the logic of all referenced models, regardless of whether their logic is up-to-date, by setting the value of GenerateUpdatedOnly in the CZ_DB_SETTINGS table to NO.

See the *Oracle Configurator Implementation Guide* for background on viewing or editing the values in the CZ_DB_SETTINGS table.

Container Model Structure



Configurator Implementation in the Solution

This section contains information about implementing Oracle Configurator in the TSO solution.

Create Configuration Model

This section explains how to define a configuration Model specifically for an Oracle TSO solution. In the procedure below, you log in to Oracle Forms with Configurator Administrator responsibility and navigate to Populate/Refresh Configuration Models, Populate Configuration Models.

Steps

1. Import the Container Model by running the Import Configuration Models concurrent program.

2. After importing the Container Model, verify that its structure is valid by opening the Model for editing in Configurator Developer and choosing Generate Logic from the General area of the Workbench.

3. Add the Model structure and define configuration rules. For transient items and attributes, set the transient flag.

4. Generate a User Interface and optionally create Activate Instance UI controls on specific UI windows.

   Important: Oracle Configurator does not provide any User Interfaces in your installation. Since User Interfaces are based on specific model structure and configuration rules, you must generate your User Interfaces with Oracle Configurator Developer, or construct your own custom UIs.

An Activate Instance UI control is a button or picture to which you assign the Activate Instance action.

Customizing the User Interface consists of adding Activate buttons, creating Connection List buttons, hiding Configurator Extensions, and altering the format in the UI Editor.

To help end users to navigate when updating an installed configuration, you may also want to display the Navigation Tree in a Container Model's User Interface.

The Activate Instance UI control appears in a UI window only if you create the control on the selected Component's UI window. When you make the Oracle Configurator UI window editable, the Activate Instance control does not appear.

5. Define Line Types in Oracle Order Management. In Oracle Order Management, Transaction Types are the source of various attributes on a sales order's header and lines. When you reconfigure an installed configuration, Oracle Configurator uses Line transaction types to describe the type of change made to each item.

6. Use the following Configurator Extensions. These Configurator Extensions should be suitable without modification, but you may adapt them if necessary.

   - **Line Type:** Identifies the Line Type for use in the Summary window, and for use in downstream quoting and fulfillment activities. The Line Type column in the Summary window lists the type of change made to each item during the reconfiguration session.

   - **Interactive Location Search:** Sets the Location ID and Location Type Code for BOM trackable root instances and passes this information to downstream applications.

   - **IBAttribute:** Collects configuration attributes of configured Components.

   See the chapter, Set Up Configurator Extensions, for more information on these extensions.

7. Publish the Container Model and specify the hosting applications that you want the Model available to (e.g., Oracle Install Base, Oracle Quoting, Oracle Order Management, or Oracle iStore) in the list of hosting applications. For a list of Oracle Applications short names, see the *Oracle Configurator Implementation Guide*. For information about publishing, see the *Oracle Configurator Developer User's Guide*.

Publish Container Model

To allow users of downstream applications to reconfigure installed configurations, you must include these applications when publishing the Model from Configurator Developer. Additionally, you must make the publication available to Oracle Order Management. For more information about publishing, see the *Oracle Configurator Developer User's Guide*.

Define Profile Options

You must define some profile options before uses of other applications can reconfigure installed configurations using Oracle Configurator. The following Oracle Configurator profile options affect how Oracle Configurator integrates with Oracle Install Base, or how Oracle Configurator saves data internally and returns it for use by the host application:

- CZ: Auto-Expire Discontinued IB Trackable Items
- CZ: Configurator Install Base
- CZ: Include Unchanged Install Base Items
- CZ: Only Create CZ Config Items for Selected Nodes
- CZ: Report All Baseline Conflicts
- CZ: Suppress Baseline Errors

These profile options have default values, but you may want to modify them for your installation. For more information, see the *Oracle Configurator Installation Guide*.

You should also set certain profile options to disable pricing, as described in Disable Pricing, page 10-21.

You must also define some additional profile options to allow other applications to integrate with Oracle Configurator. For more information, refer to:

- Set Up Oracle iStore chapter of this guide
- *Oracle Install Base Implementation Guide*
- *Oracle Quoting Implementation Guide*
- *Oracle iStore Implementation and Administration Guide*

Disable Pricing

Pricing attributes sourced from Oracle Configurator output configuration attributes are not available to the pricing engine until you have exited the Configurator by clicking Done. Therefore, if you are sourcing pricing attributes from Oracle Configurator output configuration attributes, you should disable pricing display within the Configurator.

If your application that supports reconfigurations of installed configurations includes pricing that depends on the Line Type (action) or configuration attributes of the line item, then you should disable pricing. Since such a dependency is likely to be common, it is recommended that you disable pricing in Oracle Configurator if your application

supports reconfigurations of installed configurations.

To disable pricing in Oracle Configurator, set the following profile options to the value No:

- CZ: Enable ATP
- CZ: Enable List Prices
- CZ: Enable Selling Prices

For more information on setting these profile options, see the *Oracle Configurator Installation Guide*.

Map Install Base Extended Attributes to Configuration Attributes

If you are using extended attributes with the TSO solution, you must map the following to each other for all trackable items:

- The extended attributes that you define in Oracle Install Base
- The configuration attributes that you define on your configuration Model in Configurator Developer

The result of this mapping is that the values of configuration attributes produced during a session in Oracle Configurator remain associated with their trackable items in Oracle Install Base.

You may not need to map transient attributes to Install Base Extended Attributes. You should map them if they are needed during fulfillment. See the section, "Transient Attributes", above, for background.

In the following procedure, log in to Oracle Forms with Oracle Installed Base Admin responsibility and navigate to Setups, Extended Attribute Template.

Steps

1. Define the desired extended attributes, associating them at the item group level. For information on defining Oracle Install Base attributes, see Create Extended Attributes, page 8-2 in the chapter on setting up Oracle Install Base, and the *Oracle Install Base Implementation Guide*.

2. In Oracle Configurator Developer, modify your Model to add attribute Features and Properties. The methodology for modifying your Model is described in *Oracle Configurator Methodologies*, in the section on implementing configuration attributes for output. Consult that chapter for guidance, with these essential differences:
   - Instead of descriptive flexfield segments and contexts, use Oracle Install Base extended attributes and groups.

- In place of the Property names, ATTR_CONTEXT and ATTR_*n*_CONTEXT, use ATTR_GROUP and ATTR_*n*_GROUP.

Example:

In Oracle Install Base, define an extended attribute at the Inventory Item group access level with:

- Attribute Code = PORT_SPEED
  - Attribute Name = Port Speed
  - Attribute Category = EAST

    Note: The Attribute Name must be unique, for extended attributes used in the TSO flow.

In Configurator Developer, in a trackable BOM Model, define an attribute Feature with:

- Feature Name = Port_Speed
  - Feature Options = 192 Kbps, 256 Kbps, 640 Kbps

For the Feature named Port_Speed, define a Property with:

- Property Name = ATTR_NAME
  - Property Value = PORT_SPEED

For the trackable BOM Model, define a Property with:

- Property Name = ATTR_1_PATH
  - Property Value = Port_Speed

For the trackable BOM Model, define a second Property with:

- Property Name = ATTR_1_MODE
  - Property Value = 2

For the trackable BOM Model, define a third Property with:

- Property Name = ATTR_1_GROUP
  - Property Value = EAST

Oracle Configurator Schema Customizations

For your enhanced understanding of the TSO solution, this section describes some ways in which the Oracle Configurator (CZ) schema is adapted for TSO.

DISPLAY_SUMMARY_FULFILLMENT_ACTION Parameter

Using the DISPLAY_SUMMARY_FULFILLMENT_ACTION parameter, you can control whether the Line Type column appears in the Oracle Configurator Summary window during a reconfiguration session. Following are the settings of this parameter in the CZ_DB_SETTINGS table:

- **Setting ID:** DISPLAY_SUMMARY_FULFILLMENT_ACTION
- **Section Name:** UISERVER
- **Data Type:** String
- **Default Value:** True

Note: This parameter affects only reconfiguration scenarios, not new configurations. In a new configuration, the Line Type column never appears.

If the parameter is True or is not defined, then the Line Type column appears in the Summary window, if a reconfiguration. If set to False, the Line Type column does not appear.

More information on Line Types can be found in the chapter, Set Up Configurator Extensions.

See the *Oracle Configurator Implementation Guide* for background on viewing or editing the values in the CZ_DB_SETTINGS table.

CZ_CONFIG_EXT_ATTRIBUTES Table

The table, CZ_CONFIG_EXT_ATTRIBUTES, is the intermediate store of configuration attribute data between Oracle Configurator and Oracle Install Base. Oracle Configurator writes output data to the table when the IBAttribute Configurator Extension runs. Oracle Configurator compares the attribute values written to CZ_CONFIG_EXT_ATTRIBUTES with the values of the corresponding columns in the Oracle Install Base tables.

For more information about this Configurator Extension, see the chapter, Set Up Configurator Extensions.

The following table shows the layout and description of the CZ_CONFIG_EXT_ATTRIBUTES table.

CZ_CONFIG_EXT_ATTRIBUTES

Column Name	Null?	PK?	Type	Comments
CONFIG_HDR_ID	N	Y	NUMBER	Instance Header ID. Corresponds to CZ_CONFIG_ITEMS.INSTANCE_HDR_ID.
CONFIG_REV_NBR	N	Y	NUMBER	Instance Revision Number. Corresponds to CZ_CONFIG_ITEMS.INSTANCE_REV_NBR.
CONFIG_ITEM_ID	N	Y	NUMBER	Configuration Item ID. Corresponds to CZ_CONFIG_ITEMS.CONFIG_ITEM_ID.
ATTRIBUTE_LEVEL	N	Y	VARCHAR2(255)	Corresponds to CSI_I_EXTENDED_ATTRIBS.ATTRIBUTE_LEVEL.
ATTRIBUTE_NAME	N	Y	VARCHAR2(255)	Corresponds to CSI_I_EXTENDED_ATTRIBS.ATTRIBUTE_CODE.
ATTRIBUTE_GROUP	Y	N	VARCHAR2(255)	Corresponds to CSI_I_EXTENDED_ATTRIBS.ATTRIBUTE_GROUP.
ATTRIBUTE_VALUE	Y	N	VARCHAR2(255)	Corresponds to CSI_I_EXTENDED_ATTRIBS.ATTRIBUTE_VALUE.
SEQUENCE_NBR	Y	N	NUMBER	Sequence number for operations.

The columns, CONFIG_HDR_ID, CONFIG_REV_NBR, CONFIG_ITEM_ID, ATTRIBUTE_LEVEL, and ATTRIBUTE_NAME, are the primary key for CZ_CONFIG_EXT_ATTRIBUTES.

The column SEQUENCE_NBR indicates a global sequence number for a batch validation session's processing of configuration operations and extended attributes. This sequence number is used only during the batch validation of pseudo-configurations, such as when an item is validated by Oracle Contact Center. For more details, see the section, "Batch Validation Parameters, page 10-29"

For brevity, the preceding table does not include standard columns such as LAST_UPDATE_DATE.

Mapping of Configurator Tables to Install Base Schema

When adding trackable data, the system adds the item into the Oracle Install Base table, CSI_T_TXN_LINE_DETAILS. If change occurs to a transaction value after a reconfiguration, then a comparison happens between CSI_T_TXN_LINE_DETAILS and CZ_CONFIG_ITEMS to report the change. The CSI_T_TRANSACTION_LINES table contains the transaction detail for the configured item. There is one transaction line for each source transaction.

The following table shows the mapping of the Oracle Configurator (CZ) tables to the Oracle Install Base (IB) schema.

Mapping of CZ Tables to CSI Tables

CZ_TABLE.COLUMN	CSI_TABLE.COLUMN	Description
CZ_CONFIG_ITEMS.INSTANCE_HDR_ID	CSI_T_TXN_LINE_DETAILS.CONFIG_HDR_ID	Each configured Component has its own unique instance header identification. The CONFIG_HDR_ID populates the INSTANCE_HDR_ID from the CZ_CONFIG_ITEMS table.
CZ_CONFIG_ITEMS.INSTANCE_REV_NBR	CSI_T_TXN_LINE_DETAILS.CONFIG_REV_NBR	Each configured Component has its own unique instance revision number. The CONFIG_REV_NBR populates the INSTANCE_REV_NBR from the CZ_CONFIG_ITEMS table.
CZ_CONFIG_ITEMS.CONFIG_ITEM_ID	CSI_T_TXN_LINE_DETAILS.CONFIG_ITEM_ID	This is the unique identifier for the configured item.
CZ_CONFIG_ITEMS.TARGET_HDR_ID	CSI_T_II_RELATIONSHIPS.OBJ_CONFIG_INST_HDR_ID	This field is populated only for Connectors that point to the target.

CZ_TABLE.COLUMN	CSI_TABLE.COLUMN	Description
CZ_CONFIG_ITEMS.TARGET_REV_NBR	CSI_T_II_RELATIONSHIPS.OBJ_CONFIG_REV_NBR	This field is populated only for Connectors that point to the target.
CZ_CONFIG_ITEMS.TARGET_ITEM_ID	CSI_T_II_RELATIONSHIPS.OBJ_CONFIG_ITEM_ID	This field is populated only for Connectors that point to the target.
CZ_CONFIG_ITEMS.CONFIG_ITEM_ID	CSI_T_TRANSACTION_LINES.SOURCE_TRANSACTION_ID	This is the ID of the item being configured.
CZ_CONFIG_ITEMS.UOM_CODE	CSI_T_TXN_LINE_DETAILS.UNIT_OF_MEASURE	This is the units of measurement code.
CZ_CONFIG_ITEMS.LOCATION_ID	CSI_T_TXN_LINE_DETAILS.LOCATION_ID	This is the Location ID for the item.
CZ_CONFIG_ITEMS.LOCATION_TYPE_ID	CSI_T_TXN_LINE_DETAILS.LOCATION_TYPE_CODE	This is the Location Type Code for the item.
CZ_CONFIG_HDRS_V.BASELINE_REV_NBR	CSI_ITEM_INSTANCES.CONFIG_INST_REV_NUM CSI_T_TXN_LINE_DETAILS.CONFIG_INST_BASELINE_REV_NUM	CZ_CONFIG_HDRS_V.BASELINE_REV_NBR is a reference to CSI_ITEM_INSTANCES.CONFIG_INST_REV_NUM, which is the existing instance revision in Install Base that is being reconfigured. When Oracle Configurator writes transactions for a reconfiguration, it inserts this number in both CZ_CONFIG_HDRS_V.BASELINE_REV_NBR and CSI_T_TXN_LINE_DETAILS.CONFIG_INST_BASELINE_REV_NUM.
CZ_CONFIG_EXT_ATTRIBUTES.ATTRIBUTE_NAME	CSI_T_EXTEND_ATTRIBS_V.ATTRIBUTE_CODE	The name of the extended attribute for the item. .
CZ_CONFIG_EXT_ATTRIBUTES.ATTRIBUTE_VALUE	CSI_T_EXTEND_ATTRIBS_V.ATTRIBUTE_VALUE	The value of the extended attribute for the item. .

Link Items Column

The column, IB_LINK_ITEM_FLAG, indicates whether a node is a link item. This column is part of the tables, CZ_PS_NODES and CZ_IMP_PS_NODES. A true value for this column means that the node is a link item. This data is regarded as the *link flag*. For more information on links, see the chapter, Set Up Inventory and Network Link Items, page 10-11.

See the chapter, Configurator Extensions, for information on how the CIO uses the link flag.

For table details, see the CZ *e*TRM on MetaLink, Oracle's technical support Web site.

Transient Items Column

The column, TRANSIENT_FLAG, indicates whether a node is a transient item. This column is part of the table, CZ_PS_NODES. A true value for this column means that the node is a transient item. This data is regarded as the *transient flag*.

For table details, see the CZ *e*TRM on MetaLink, Oracle's technical support Web site.

Tangible Items Columns

The columns SHIPPABLE_ITEM_FLAG, INVENTORY_TRANSACTABLE_FLAG, SERIALIZABLE_ITEM_FLAG, and ASSEMBLE_TO_ORDER_FLAG indicate whether a node is a tangible item. These columns are part of the tables CZ_PS_NODES and CZ_IMP_PS_NODES.

The columns TANGIBLE_ITEM_FLAG and RETURNED_FLAG indicate whether a node is a tangible item that has been returned. These columns are part of the table CZ_CONFIG_ITEMS.

For table details, see the CZ *e*TRM on MetaLink, Oracle's technical support Web site.

Initialization Parameters

This section describes parameters that are specific to the TSO solution.

For information on the use of parameters in the initialization message, see the *Oracle Configurator Implementation Guide*.

When configuring instances of trackable Components prior to their installation in Oracle Install Base, the system uses the following parameters:

- instance
- validation_context

For background on updating configuration instances, see the chapter, Oracle TSO Business Process, page 2-1.

instance

The system uses the `instance` parameter when configuring instances of trackable Components. This parameter lets you configure multiple instances in the same configuration session. In a host application, this corresponds to the end user's ability to select multiple Components for configuration.

This is the only initialization parameter that can be specified multiple times in the initialization message.

In order to specify each instance that requires configuration, provide the Configuration Header ID (CZ_CONFIG_DETAILS_V.CONFIG_HDR_ID) as the value of the `header_id` sub-parameter. For example:

```
<instance header_id="17848"/>
<instance header_id="18333"/>
<instance header_id="23445"/>
```

validation_context

The system uses the `validation_context` parameter when configuring instances of trackable configuration Components. The value sets the context in which the following occur:

- Validation of the current instances
- Comparison of the current instances to the installed instances in Oracle Install Base

Batch validation uses the `validation_context` parameter. For more information, see the section, Batch Validation Parameters, page 10-29.

The only allowed value for the validation_context parameter is INSTALLED. This value compares the revisions of the current instances only to the revisions of those instances that have been installed in Oracle Install Base.

Batch Validation Parameters

For background and details about batch validation, see the *Oracle Configurator Implementation Guide*.

TSO implementers must use one of the following optional values for the `p_validation_type` parameter to the CZ_CF_API.VALIDATE procedure. (For details on the VALIDATE procedure used by batch validation, see VALIDATE Procedure, page 10-36.)

Values are:

- `validate_order`: The process should pass this value when validating orders, such as what Oracle Order Management does. This is the default value.
- `validate_fulfillment`: The process should pass this value when validating fulfillment status, such as what Oracle Install Base does.

Example

```
<batch_validate validation_type="validate_order">
```

Programmatic Tools for TSO Instance Management

You can use certain programmatic tools to help manage TSO instance configurations. This section describes the following tools or procedures modified for use with the TSO solution:

- COPY_CONFIGURATION, page 10-30
- COPY_CONFIGURATION_AUTO, page 10-36
- VALIDATE, page 10-36

For full background and details about these programmatic tools for development, see the *Oracle Configurator Implementation Guide*.

The versions of these procedures described in this section are adapted to operate on configurations that have connectivity, as well as on those that do not.

These procedures use the custom data type, CZ_API_PUB.NUMBER_TBL_TYPE, which is a table of number type. For background on custom data types, see the appropriate references in the *Oracle Configurator Implementation Guide* and in Custom Data Types in the Package CZ_CF_API, page 10-39.

COPY_CONFIGURATION Procedure

This procedure, COPY_CONFIGURATION in the CZ_CONFIG_API_PUB package, makes a copy of a saved configuration in the Oracle Configurator schema.

This procedure, created for use with the TSO solution, is a variation of COPY_CONFIGURATION in the CZ_CF_API package. For a description of the CZ_CF_API package, see the *Oracle Configurator Implementation Guide*. The most important difference is that this TSO procedure has two additional OUT parameters: `x_orig_item_id_tbl` and `x_new_item_id_tbl`. These parameters inform host applications about changes to the Configuration Item IDs (CZ_CONFIG_ITEMS.CONFIG_ITEM_ID) in the configuration in a case where you are copying Component instances that are not yet installed in Oracle Install Base. These parameters are of type CZ_API_PUB.NUMBER_TBL_TYPE, which is a table of the NUMBER type. As used by the COPY_CONFIGURATION procedure, these output tables are indexed from 1 to *n*, where *n* is the number of Configuration Item IDs changed from the source configuration to the new configuration. You can use the index number of the IDs stored in the tables to map the Configuration Item IDs in the original configuration to the Configuration Item IDs in the new copy of that configuration. The following example shows how a host application might process these outputs using Oracle Configurator.

Example: Mapping Original Configuration Item IDs to New IDs When Copying a Configuration

```
...
FORALL i IN x_orig_item_id_tbl.FIRST..x_orig_item_id_tbl.LAST
  UPDATE my_order_lines
  SET config_item_id = x_new_item_id_tbl(i)
  WHERE my_order_header = l_header
    AND config_hdr_id = x_config_hdr_id
    AND config_rev_nbr = x_config_rev_nbr
    AND config_item_id = x_orig_item_id_tbl(i);
...
```

The non-TSO version of COPY_CONFIGURATION, in the package CZ_CF_API, has been modified to create an exception if it generates any new Configuration Item IDs. This averts data corruption in Oracle Order Management, Order Capture, and Order Contracts. If you encounter this exception, you must apply the appropriate patch for your host application that invokes the new COPY_CONFIGURATION procedure from CZ_CONFIG_API_PUB.

For background on configurations, see the chapter on managing configurations in the *Oracle Configurator Implementation Guide*.

Considerations Before Running

Keep the following considerations in mind when running COPY_CONFIGURATION.

Prerequisites

The configuration that you are copying must exist. If the configuration has been logically deleted, then you can pass 0 as the value of `p_handle_deleted_flag` to enable copying.

Timing

You should use this procedure every time you copy a configuration. The procedure ensures copying of all inputs, outputs, attributes, and messages.

Warnings

If the configuration does not exist, or if the copy fails, `x_msg_count` will be greater than zero, and `x_msg_data` will contain error information. Check `x_return_status` for the return status.

Syntax and Parameters

The syntax for this procedure is:

```
PROCEDURE copy_configuration(p_api_version IN  NUMBER
    ,p_config_hdr_id        IN  NUMBER
    ,p_config_rev_nbr       IN  NUMBER
    ,p_copy_mode            IN  VARCHAR2
    ,x_config_hdr_id        OUT NOCOPY  NUMBER
    ,x_config_rev_nbr       OUT NOCOPY  NUMBER
    ,x_orig_item_id_tbl     OUT NOCOPY  CZ_API_PUB.number_tbl_type
    ,x_new_item_id_tbl      OUT NOCOPY  CZ_API_PUB.number_tbl_type
    ,x_return_status        OUT NOCOPY  VARCHAR2
    ,x_msg_count            OUT NOCOPY  NUMBER
    ,x_msg_data             OUT NOCOPY  VARCHAR2
    ,p_handle_deleted_flag  IN  VARCHAR2 := NULL
    ,p_new_name             IN  VARCHAR2 := NULL
);
```

The following table describes the parameters for the COPY_CONFIGURATION procedure.

Parameters for the COPY_CONFIGURATION Procedure

Parameter	Data Type	Mode	Note
p_api_version	number	in	Required. See the description of API Version Numbers in the *Oracle Configurator Implementation Guide*.
p_config_hdr_id	number	in	Required. Specifies the Configuration Header ID of the source configuration to copy.
p_config_rev_nbr	number	in	Required. Specifies the Configuration Revision Number of the source configuration to copy.

Parameter	Data Type	Mode	Note
p_copy_mode	varchar2	in	Required. Flag that specifies whether creating a new Configuration Header ID or new Configuration Revision Number has one of the following values: • CZ_API_PUB.G_NEW_HEADER_COPY_MODE, which creates a new Configuration Header ID. Used in the case of reordering. Equivalent to the value **new_config** for the initialization parameter **save_config_behavior**. • CZ_API_PUB.G_NEW_REVISION_COPY_MODE, which creates a new Configuration Revision Number. Used in the case of reconfiguring or repricing. Equivalent to the value **new_revision** for the initialization parameter **save_config_behavior**.

Parameter	Data Type	Mode	Note
x_config_hdr_id	number	out	Identifies the Configuration Header ID of the new copy of the source configuration.
x_config_rev_nbr	number	out	Identifies the Configuration Revision Number of the new copy of the source configuration.
x_orig_item_id_tbl	CZ_API_PUB.number_tbl_type	out	Indexed table of Configuration Item IDs from the source configuration that were changed when being copied to the new configuration. Empty if the configuration does not contain trackable instances.
x_new_item_id_tbl	CZ_API_PUB.number_tbl_type	out	Indexed table of Configuration Item IDs from the new copy of the source configuration that were changed when being copied from the source configuration. Empty if the configuration does not contain trackable instances.
x_return_status	varchar2	out	Standard out parameter. Value is: FND_API.G_RET_STS_SUCCESS, FND_API.G_RET_STS_ERROR, or FND_API.G_RET_STS_UNEXP_ERROR.

Parameter	Data Type	Mode	Note
x_msg_count	number	out	Standard out parameter.
x_msg_data	varchar2	out	Standard out parameter.
p_handle_deleted_flag	varchar2	in	Specifies how to handle an attempt to copy a logically deleted source configuration. If the source configuration is logically deleted, and the value passed to this parameter is 0, then the source configuration is undeleted so that the copy can proceed. If the value passed is null, then an error is raised by the attempt to copy.
p_new_name	varchar2	in	The new name applied to the copied configuration.

Considerations After Running

Keep the following considerations in mind after running COPY_CONFIGURATION.

Results

This procedure copies all database records associated with a configuration to a new Configuration Header ID and Configuration Revision Number. The changes that the procedure made are not automatically committed. When you call this procedure, you must check `x_return_status` and commit the transaction if the copy was successful.

Troubleshooting

Examine `x_return_status`. If it does not indicate success, then examine `x_msg_count` for the number of error messages, and `x_msg_data` for the error messages themselves.

COPY_CONFIGURATION_AUTO Procedure

This procedure runs COPY_CONFIGURATION within an autonomous transaction. If the copy is successful, new data will be committed to the database without impacting the caller's transaction.

VALIDATE Procedure

This section describes a new parameter signature for the VALIDATE procedure in the CZ_CF_API package. For a description of the previous signature for the VALIDATE procedure, see the *Oracle Configurator Implementation Guide*. Existing calls to the previous signature will continue to operate correctly, and do not need to be changed.

For TSO, you must call this procedure with the parameter, `p_validation_type`. When Oracle Configurator calls this procedure during batch validation, it uses the following values for that parameter:

- CZ_API_PUB.VALIDATE_ORDER: This value should be passed when validating orders, such as what Oracle Order Management does. This is the default value.
- CZ_API_PUB.VALIDATE_FULFILLMENT: This value should be passed when validating fulfillment status, such as what Oracle Install Base does.
-

Syntax and Parameters

The syntax for this procedure is:

```
procedure validate(p_api_version          IN  NUMBER
                  ,p_config_item_tbl      IN  config_item_tbl_type
                  ,p_config_ext_attr_tbl IN  config_ext_attr_tbl_type
                  ,p_url                  IN  VARCHAR2
                  ,p_init_msg             IN  VARCHAR2
                  ,p_validation_type      IN  VARCHAR2
                  ,x_config_xml_msg   OUT NOCOPY CFG_OUTPUT_PIECES
                  ,x_return_status    OUT NOCOPY VARCHAR2
                  ,x_msg_count        OUT NOCOPY NUMBER
                  ,x_msg_data         OUT NOCOPY VARCHAR2
                  );
```

The following table describes the parameters for the new parameter signature of the VALIDATE procedure. Several parameters are new, as indicated. Parameters retained from the previous signature are described in more detail in the *Oracle Configurator Implementation Guide*

The parameter `p_validation_type` has special meaning for the TSO solution, as noted elsewhere.

Parameters for the VALIDATE Procedure

Parameter	Data Type	Mode	Note
p_api_version	number	in	Required. See the description of API Version Numbers in the *Oracle Configurator Implementation Guide*.
p_config_item_tbl (new)	config_item_tbl_type (new) See Custom Data Types in the Package CZ_CF_API, page 10-39	in	List of input selections. Can be null if only updating attributes. Individual field can be null, FND_API.G_MISS_NUM or FND_API.G_MISS_CHAR, or another value
p_config_ext_attr_tbl (new)	config_ext_attr_tbl_type (new) See Custom Data Types in the Package CZ_CF_API, page 10-39	in	List of extended attributes to be updated.
p_url	varchar2	in	The URL of the Oracle Configurator Servlet. If null, the value of the profile option CZ_UIMGR_URL is used. Same as previous parameter `url`

Parameter	Data Type	Mode	Note
p_init_msg	varchar2(2000)	in	The XML initialization message for Oracle Configurator. Cannot be null. Same as previous parameter `init_message`
p_validation_type	varchar2	in	Validation type. The possible values are CZ_API_PUB.VALIDATE_ORDER, CZ_API_PUB.VALIDATE_FULFILLMENT, and CZ_API_PUB.INTERACTIVE. The default is CZ_API_PUB.VALIDATE_ORDER.
x_config_xml_msg	CFG_OUTPUT_PIECES	out nocopy	A table of the output XML messages produced by validating the configuration. Same as previous parameter `config_messages`
x_return_status (new)	varchar2	out nocopy	Standard out parameter. Value is: FND_API.G_RET_STS_SUCCESS, FND_API.G_RET_STS_ERROR, or FND_API.G_RET_STS_UNEXP_ERROR.
x_msg_count (new)	number	out nocopy	Standard out parameter. Number of log messages.

Parameter	Data Type	Mode	Note
x_msg_data (new)	varchar2	out nocopy	Standard out parameter. If x_msg_count equals 1, contains message.

This version of the VALIDATE procedure uses standard Oracle Applications (FND) logging. If logging is enabled, the following information will be written into the log:

- Input selections (statement level)
- External attributes (statement level)
- Final return status of the batch validation session (procedure level)
- New configuration info generated in the batch validation session (procedure level)
- Error message(s) (procedure level)

Some parameters of the VALIDATE procedure use custom data types. These are described in Custom Data Types in the Package CZ_CF_API, page 10-39. For background, see the section on Custom Data Types in the *Oracle Configurator Implementation Guide*.

Custom Data Types in the Package CZ_CF_API

Custom Type	Description
config_item_tbl_type	Table of config_item_rec_type, indexed by BINARY_INTEGER
config_ext_attr_tbl_type	Table of config_ext_attr_rec_type, indexed by BINARY_INTEGER

Custom Type	Description
config_item_rec_type	Record consisting of: config_item_id cz_config_items.config_item_id%TYPE component_code cz_config_items.node_identifier%TYPE sequence_nbr cz_config_items.sequence_nbr%TYPE operation NUMBER quantity cz_config_items.item_num_val%TYPE instance_name cz_config_items.name%TYPE location_id cz_config_items.location_id%TYPE location_type_code cz_config_items.location_type_code%TYPE
config_ext_attr_rec_type	Record consisting of: config_item_id cz_config_ext_attributes.config_item_id%TYPE component_code VARCHAR2(1200) sequence_nbr NUMBER attribute_name cz_config_ext_attributes.attribute_name%TYPE attribute_group cz_config_ext_attributes.attribute_group%TYPE attribute_value cz_config_ext_attributes.attribute_value%TYPE
Constants	BV_OPERATION_UPDATE CONSTANT NUMBER := 1; BV_OPERATION_DELETE CONSTANT NUMBER := 2;

For an example of how you might call the VALIDATE procedure, see Calling the VALIDATE Procedure, page 10-40. This example is *not* ready to run as shown. You must replace some of the placeholder values in it to fit your own implementation.

Calling the VALIDATE Procedure

```
-- sample_bvapicall.sql
set serveroutput on;
DECLARE
  l_config_item_rec CZ_CF_API.config_item_rec_type;
  l_config_item_tbl CZ_CF_API.config_item_tbl_type;
  l_config_attr_rec  CZ_CF_API.config_ext_attr_rec_type;
  l_config_attr_tbl  CZ_CF_API.config_ext_attr_tbl_type;
  l_url              VARCHAR2(100);
  l_init_msg         VARCHAR2(2000);
  l_validation_type  VARCHAR2(1) := CZ_API_PUB.VALIDATE_ORDER;
  l_config_xml_msg   CZ_CF_API.CFG_OUTPUT_PIECES;
  l_return_status    VARCHAR2(1);
  l_msg_count        NUMBER;
  l_msg_data         VARCHAR2(1000);

  l_user_id number := 1111; -- your user id
  l_resp_id number := 2222; -- responsibility id
  l_appl_id number := 3333; -- application id

BEGIN
  l_url :=
'http://www.mysite.com:myportnumber/configurator/oracle.apps.cz.servlet.
UiServlet';
  l_init_msg :=
    '<initialize>'
      || '<param name="database_id">mydbhost_mydbname</param>'
      || '<param name="responsibility_id">2222</param>'
      || '<param name="calling_application_id">3333</param>'
      || '<param name="context_org_id">4444</param>'
      || '<param name="model_id">5555</param>'
      || '<param
name="config_creation_date">'||to_char(sysdate)||'</param>'
      || '<param name="config_header_id">6666</param>'
      || '<param name="config_rev_nbr">1</param>'
      || '<param name="save_config_behavior">new_revision</param>'
      || '<param name="read_only">FALSE</param>'
      || '<param name="terminate_msg_behavior">brief</param>' ||
    '</initialize>';

  -- update instance name
  l_config_item_rec.config_item_id := 1000;
  l_config_item_rec.component_code := '123-456';
  l_config_item_rec.operation := CZ_CF_API.bv_operation_update;
  l_config_item_rec.sequence_nbr := 1;
  l_config_item_rec.instance_name := 'new instance name';
  l_config_item_tbl(l_config_item_tbl.count+1) := l_config_item_rec;

  -- clear out location_id and location_type_code
  l_config_item_rec := null;
  l_config_item_rec.config_item_id := 1011;
  l_config_item_rec.component_code := '123-456-789';
  l_config_item_rec.operation := CZ_CF_API.bv_operation_update;
  l_config_item_rec.sequence_nbr := 3;
  l_config_item_rec.location_id := fnd_api.G_MISS_NUM;
  l_config_item_rec.location_type_code := fnd_api.G_MISS_NUM;
  l_config_item_tbl(l_config_item_tbl.count+1) := l_config_item_rec;

  -- delete an instance
  l_config_item_rec := null;
  l_config_item_rec.config_item_id := 1001;
  l_config_item_rec.component_code := '123-456';
```

```
l_config_item_rec.operation := CZ_CF_API.bv_operation_delete;
  l_config_item_rec.sequence_nbr := 4;
  l_config_item_tbl(l_config_item_tbl.count+1) := l_config_item_rec;

  l_config_attr_rec.config_item_id := 1010;
  l_config_attr_rec.component_code := '123-456-678';
  l_config_attr_rec.sequence_nbr := 2;
  l_config_attr_rec.attribute_name := 'an attr name';
  l_config_attr_rec.attribute_group := 'an attr group';
  l_config_attr_rec.attribute_value := 'a new attr value';
  l_config_attr_tbl(l_config_attr_tbl.count+1) := l_config_attr_rec;

  fnd_global.apps_initialize(1111,2222,3333);
  CZ_CF_API.VALIDATE(1.0
                    ,l_config_item_tbl
                    ,l_config_attr_tbl
                    ,l_url
                    ,l_init_msg
                    ,l_validation_type
                    ,l_config_xml_msg
                    ,l_return_status
                    ,l_msg_count
                    ,l_msg_data
                    );

  dbms_output.put_line('Return status: ' || l_return_status);
  if l_return_status = fnd_api.G_RET_STS_SUCCESS then
    -- parse xml output msg
  else
    dbms_output.put_line('message count: ' || l_msg_count);
    if  l_msg_count = 1 then
      dbms_output.put_line('message: ' || l_msg_data);
    else
      for i in 1 .. l_msg_count loop
        dbms_output.put_line('message ' || i || ': '
                    || fnd_msg_pub.get(i,fnd_api.g_false));
      end loop;
    end if;
  end if;
exception
  when others then
    dbms_output.put_line('Unexpected err: ' || SQLERRM);
end;
/
```

11

Set Up Oracle Configurator Extensions

This chapter covers the following topics:

- Overview of Set Up Oracle Configurator Extensions Chapter
- General Setup for Configurator Extensions
- Interactive Location Search Configurator Extension
- Line Type Configurator Extension
- IBAttribute Configurator Extension
- Using the Oracle Configuration Interface Object (CIO)

Overview of Set Up Oracle Configurator Extensions Chapter

This chapter describes setup tasks that you must perform to Oracle Configurator that are specific to the Oracle Telecommunications Service Ordering (TSO) solution.

Before You Begin

This chapter explains:

- The role of Configurator Extensions in a TSO solution and how to modify these Configurator Extensions where necessary for your requirements. Without modification, these Configurator Extensions can provide a certain amount of TSO functionality. Other optional functionality requires you to modify the Configurator Extensions.

- How to use certain features of the Oracle Configuration Interface Object (CIO) that are specifically designed to support a TSO solution.

Before you can set up Oracle Configurator Extensions for use with the Oracle TSO solution, you must set up Oracle Bills of Material, Oracle Installed Base, and Oracle Order Management. For more information, see the following chapters:

- Set Up Bills of Material
- Set Up Installed Base
- Set Up Order Management

In addition to this chapter, there is another chapter on setting up Oracle Configurator for use with the TSO solution, Set Up Oracle Configurator and Customize the Solution, page 10-1.

Setup Checklist

Following is the setup checklist for implementing TSO-specific functionality for Oracle Configurator Extensions:

Oracle Configurator Extensions-TSO Setup Checklist

Setup Step	Required/Optional	Comment
Implement Oracle Configurator	Required	You must install Oracle Configurator before implementing Configurator Extensions
Perform General Setup for Configurator Extensions, page 11-3	Required	You must perform certain tasks when setting up any of the Configurator Extensions.
Set Up **Interactive Location Search** Configurator Extension (sets the Location of configured Components)	Optional (see Comment)	An item instance must have a location assigned before it can be ordered. The provided Extension is an optional design for fulfilling that general requirement.
Set Up **Line Type** Configurator Extension (sets the Line Type of a configured Component)	Required	The provided Extension should satisfy the needs of all users, but you may replace it with one of your own design.
Set Up **IBAttribute** Configurator Extension (collects configuration attributes of a configured Component)	Optional (see Comment)	This Extension is not required if no attributes are being collected from your configured item instance, but this is not a likely implementation.

Overview of Configurator Extensions in the TSO Solution

Reconfiguring installed configurations of container Models relies on the use of one or more Configurator Extensions, depending on how much you want to customize the TSO solution, if at all.

The available solutions are:

- Default solutions, which require modest implementation work on your part
- Customized solutions, which require additional design and custom implementation to achieve your desired results. The provided solutions are sufficient for demonstrations using the Oracle Vision instance. If the business logic and behavior of the default solution does not meet your business needs, you should consider modifying or replacing the extension with your own customized version.

For more information about reconfiguring installed configurations of Container Models, see the chapter Oracle TSO Business Processes, page 2-1.

General Setup for Configurator Extensions

Some setup considerations that affect all the TSO Configurator Extensions are described in this section.

- If you do *not* need to customize a given Configurator Extension, then you can use the Java class for that extension that is already installed with Oracle Configurator. See Installed Classes for TSO Configurator Extensions, page 11-4.
- If you *do* need to modify the default behavior of any of these Configurator Extensions, then see Using Customizable Source Code for TSO Configurator Extensions, page 11-4.
- For each Configurator Extension, whether or not you customize its behavior, you must perform the associated setup procedures specific to that extension, as described in the other sections of this chapter:
  - Interactive Location Search Configurator Extension, page 11-6
  - Line Type Configurator Extension, page 11-13
  - IBAttribute Configurator Extension, page 11-26

  You will need to know how to incorporate Configurator Extensions into your configuration Model and make them available to the runtime Oracle Configurator. For this information, see the *Oracle Configurator Developer User's Guide*.

Using Installed Classes for TSO Configurator Extensions

Compiled class files for the default versions of the TSO Configurator Extensions are available directly in the class path used by the runtime Oracle Configurator. The names of the installed Java classes for the TSO Configurator Extensions are listed in the following table, along with the name of the Java package that contains them.

Installed Classes for TSO Configurator Extensions

Configurator Extension	Installed Java Package/Class
package for all	`oracle.apps.cz.cx.tso`
Interactive Location Search	`MaintainLocationCX`
Line Type	`AssignItemLineTypeCX`
IBAttribute	`CZIBAttributeCX`

If you do not need to modify the default behavior of an extension, then, when you are defining its associated Configurator Extension Rules, you do not have to compile the Java source code and place it in a Configurator Extension Archive. Instead, when you are defining a Configurator Extension Rule in Oracle Configurator Developer, simply enter the fully qualified Java class name, as text, in the field labelled **Java Class**. For example, to use the Interactive Location Search Configurator Extension in a rule, you would enter the following in the Java Class field:

Example

```
oracle.apps.cz.cx.tso.MaintainLocationCX
```

If you enter a class name for the Configurator Extension Rule directly, as described here, do not use the Choose Class button to select a class.

Using Customizable Source Code for TSO Configurator Extensions

If you need to modify the default behavior of any of the TSO Configurator Extensions, then you must modify and recompile its customizable Java source code. Begin by consulting the Note on MetaLink, Oracle's technical support Web site, that has the Subject indicated in the table Customizable Source Code for TSO Configurator Extensions, page 11-5 for "all" TSO Configurator Extensions. That Note is the starting point for obtaining the source code, and references all of the other individual MetaLink Notes that contain the source code and certain detailed setup information for the TSO Configurator Extensions. The table here lists the full MetaLink Note Subject for each Extension. You can locate each Note by searching MetaLink for this Subject string.

Customizable Source Code for TSO Configurator Extensions

Configurator Extension	MetaLink Note Subject
all	R12 Source Code for Oracle Telecommunications Service Ordering (TSO) Configurator Extensions
Interactive Location Search	R12 TSO Configurator Extension MaintainLocationCX.java (and its References)
Line Type	R12 TSO Configurator Extension AssignItemLineTypeCX.java (and its References)
IBAttribute	R12 TSO Configurator Extension CZIBAttributesCX.java

Procedure

1. Consult the MetaLink note for the appropriate extension, as listed in the table Customizable Source Code for TSO Configurator Extensions, page 11-5. Copy the text from the Sample Code field into a text file with the appropriate name.

2. Edit the Java source file for the Configurator Extension, to modify the behavior of the Java class that implements the extension.

   - The modifications that you might be likely to make to each extension are suggested elsewhere in this chapter. For instance, see Modifying the IBAttribute Configurator Extension, page 11-28.

   - For information on how to write and compile Configurator Extensions, see the *Oracle Configurator Extensions and Interface Object Developer's Guide*.

   - See Using the Oracle Configuration Interface Object (CIO), page 11-29 for information on using the API for programmatic access to the runtime Oracle Configurator.

   - The Javadoc-format comments in the source code provide details on how to bind the methods of the Java classes when defining Configurator Extension Rules.

   - Any modifications to the source should be made in a package or class different from that of the installed classes. Do not place the modified versions in a package named `oracle.apps.cz.cx.tso`, or they will be superseded at

runtime by the installed versions described in Installed Classes for TSO Configurator Extensions, page 11-4, because the Configurator Extension Archives in which you place customized classes are *appended* to the runtime class path. As a convenience, each source code sample has already been put in a different package (`oracle.apps.cz.cx.sample.tso`), to avoid conflicts with the installed class definitions.

3. Compile the Java class. See the *Oracle Configurator Extensions and Interface Object Developer's Guide* for information about compiling Configurator Extensions.

4. Place the compiled Java class in a Java class archive file (using a valid zip or JAR format) with a name of your choice.

5. In Oracle Configurator Developer, create a Configurator Extension Archive for the class. Add the Configurator Extension archive you created to the archive path for your Model. This provides your Model with access to the Java class in the archive. For details on how to do this, see the *Oracle Configurator Developer User's Guide*.

Interactive Location Search Configurator Extension

The Interactive Location Search Configurator Extension demonstrates how to interact with an Oracle Applications Framework page to display locations from an Oracle Trading Community Architecture (TCA) account, and to use the location selected by the end user to set the required Location ID and Location Type Code on a trackable Component instance. To understand the need for the Location ID and Location Type Code, see Locations, page 11-6.

This extension, or some alternative that provides location information, is required for implementing the reconfiguring of installed configurations of container Models.

> **Note:** Implementing this extension assumes that you are using a user interface generated with the HTML-based version of Oracle Configurator Developer. If you want to use a DHTML user interface, you must implement a Functional Companion, as described in a previous release of this document. Functional Companions do not have all the functionality of Configurator Extensions.

Locations

Oracle Sales lets you create multiple addresses for customers. Each address is identified by a Location ID, which is not displayed to the Oracle Sales end user. At the other end of the quote-to-installation flow, Oracle Installed Base requires that every installed Component instance have a Location ID, to identify the location at which an item has been installed. Along the course of this flow, the Location Type Code points to the database table containing the detailed information for the address that the Location ID

identifies.

An Oracle Configurator configuration Model normally represents only the configurable aspects of a Component, and thus does not have any innate means of expressing the location of a configured Component. Consequently, there is no built-in equivalent to the Location ID. For a trackable Component to be configured and later accepted by Oracle Installed Base, a Location ID must be assigned to every configured Component.

Oracle Configurator forces trackable item instances to be incomplete until the location is set, via the CIO. This incompleteness enforces the need for the location field in Oracle Installed Base. When an end user adds an instance of a trackable item, a location validation failure occurs, informing the user that the component lacks the location information required for ordering. This requirement for a location does not apply to items that are network Links. See Network Link Items, page 10-11 for details.

You must implement a Configurator Extension to assign a location to an instance, but you do not necessarily require the Interactive Location Search Configurator Extension described here. If the quote or order contains the location information, then you can use a simpler Configurator Extension to query the quote or order for the needed location data, and assign it programmatically. Oracle does not currently supply a Configurator Extension to perform these tasks.

Setting Up Interactive Location Search Configurator Extension

For the full source code of the Interactive Location Search Configurator Extension, consult MetaLink, Oracle's technical support Web site.

The following sections describe what you must do before using the Interactive Location Search Configurator Extension.

Note: Since this extension is constructed with Oracle Applications Framework, you must use the business logic written into the Java source for the controller object to modify its behavior. This requires expertise in programming for the Oracle Applications Framework.

General Setup

Perform the following general setups.

1. In Oracle Trading Community Architecture (TCA), perform the required tasks for setting up Locations and Location IDs.

2. In Oracle Sales, define your customers.

Deploy the Oracle Applications Framework Files

If you are customizing the Configurator Extension, then you may also need to deploy customized versions of the Oracle Applications Framework files for the Configurator Extension. If you are using the installed classes for the Configurator Extension, then

skip this section.

1. Obtain the Oracle Applications Framework files for Interactive Location Search from MetaLink, Oracle's technical support Web site.

2. Customize the files as desired, using Oracle JDeveloper. See the section Modify the Interactive Location Search Configurator Extension, page 11-13, for more information.

   Warning: To preserve your customizations, give the files your own variations on their packages or names. Otherwise they will be overwritten when you install patches for Oracle Configurator.

Define the Configurator Extension Rule

Important: Be sure to check the MetaLink Note for this Configurator Extension, as listed in Customizable Source Code for TSO Configurator Extensions, page 11-5, to see whether there have been updates to the source code or rule binding instructions.

You need to define Configurator Extension Rules in Oracle Configurator Developer for use with the Interactive Location Search Configurator Extension. For details about Configurator Extension archives and defining Configurator Extension Rules, see the *Oracle Configurator Developer User's Guide*.

1. If you are customizing the Configurator Extension, perform the steps under Using Customizable Source Code for TSO Configurator Extensions, page 11-4.

2. For each trackable root Model in your container Model that requires a location, define one Configurator Extension Rule having three bindings, as described in the following steps:

   1. Define a Configurator Extension Rule with the options listed below.

      - **Name:** Enter `Select Location`. (This is a suggested convention. You may choose any name.)

      - **Model Node:** Choose the trackable root node of your Model whose instances need their Location set.

      - **Java Class:** Specify `oracle.apps.cz.cx.tso.MaintainLocationCX`.

      - **Java Class Instantiation:** Choose `With Model Node Instance`.

   2. For the Configurator Extension Rule described above, create an **event binding**,

using the options listed below:

- **Event:** Choose `onCommand`.
- **Command Name:** Enter a string that you want to use as a command. For example: `Pick Locations`. Do not enclose the string in quotation marks. The string can contain spaces The string becomes the default title for the generated UI button that runs the Configurator Extension.
- **Event Scope:** Choose `Base Node`.
- **Method:** Choose `redirectLocationSearch(HttpServletResponse Arg1, oracle.apps.cz.cio.BomModel Arg2)`.

3. For the event binding shown above, create **argument bindings**, using the options listed below:

   Argument Binding 1

   - **Argument Name:** Accept `response` or `Arg1`.
   - **Argument Type:** Accept `javax.servlet.http.HttpServletResponse`.
   - **Argument Specification:** Choose `Event Parameter`.
   - **Binding:** Choose `HttpServletResponse`.

   Argument Binding 2

   - **Argument Name:** Accept `trackableRoot` or `Arg2`.
   - **Argument Type:** Accept `oracle.apps.cz.cio.BomModel`.
   - **Argument Specification:** Choose Model Node or Property.
   - **Binding:** Choose the trackable root BomModel.

4. For the Configurator Extension Rule described above, create another **event binding**, using the following options:

   - **Event:** Choose `onCommand`.
   - **Command Name:** Enter the exact string `updateLocationData` as the command. This name is mandatory for this binding to function correctly.
   - **Event Scope:** Choose `Base Node`.
   - **Method:** Choose

```
updateLocationData(oracle.apps.cz.cio.BomModel Arg1,
oracle.apps.cz.cio.CXEvent Arg2)or
updateLocationData(oracle.apps.cz.cio.BomModel Arg1,
oracle.apps.cz.cio.TextFeature Arg2,
oracle.apps.cz.cio.CXEvent Arg3).
```

> **Note:** If you wish to implement the alternate version of the `updateLocationData()` method, which updates a TextFeature with the current TCA address, then you would choose the signature of `updateLocationData()` that includes a TextFeature parameter, and then define an argument binding for that parameter, as shown below under "(Optional) Argument Binding 2".

5. For the above event binding, create **argument bindings**, using the options listed below:

   Argument Binding 1

   - **Argument Name:** Accept `trackableRoot` or `Arg1`.
   - **Argument Type:** Accept `oracle.apps.cz.cio.BomModel`.
   - **Argument Specification:** Choose Model Node or Property.
   - **Binding:** Choose the trackable root BomModel.

   (Optional) Argument Binding 2

   - **Argument Name:** Accept `addressFeature` or `Arg2`.
   - **Argument Type:** Accept `oracle.apps.cz.cio.TextFeature`.
   - **Argument Specification:** Choose Model Node or Property.
   - **Binding:** Choose the Text Feature used to show the selected address to end users.

   Argument Binding 2 (or 3)

   - **Argument Name:** Accept `commandEvent` or `Arg2` (or `Arg3`).
   - **Argument Type:** Accept `oracle.apps.cz.cio.CXEvent`.
   - **Argument Specification:** Choose System Parameter.
   - **Binding:** CXEvent.

6. If it is a practice of your organization to allow changes to the TCA address data for a location between reconfigurations, then you can optionally synchronize the Text Feature with TCA, using the `updateCurrentAddress()` method. Otherwise, skip this binding.

   Given the preceding information, then for the Configurator Extension Rule described above, optionally create an **event binding**, using the following options:

   - **Event:** Choose `postInstanceLoad`.
   - **Event Scope:** Choose `Base Node`.
   - **Method:** Choose `updateCurrentAddress(oracle.apps.cz.cio.BomModel Arg1, oracle.apps.cz.cio.TextFeature Arg2)`.

7. For the above event binding, create **argument bindings**, using the options listed below:

   Argument Binding 1

   - **Argument Name:** Accept `trackableRoot` or `Arg1`.
   - **Argument Type:** Accept `oracle.apps.cz.cio.BomModel`.
   - **Argument Specification:** Choose Model Node or Property.
   - **Binding:** Choose the trackable root BomModel.

   Argument Binding 2

   - **Argument Name:** Accept `currentAddress` or `Arg2`.
   - **Argument Type:** Accept `oracle.apps.cz.cio.TextFeature`.
   - **Argument Specification:** Choose Model Node or Property.
   - **Binding:** Choose the Text feature that displays the current address obtained from TCA.

     This is the same Text Feature that you previously used in the binding for `updateLocationData()`. The `updateLocationData()` method sets the Text Feature after the Oracle Applications Framework search. The `updateCurrentAddress()` method corrects the value if the TCA data has changed, since the Text Feature would have an incorrect text value for the old data

3. After you have defined the event and argument bindings for the items requiring

locations, generate logic for your Model in order to reflect the addition of the Configurator Extension Rules.

4. Create or refresh a user interface for your Model. This creates two buttons in the user interface that by default are captioned with the Command Names that you specified in the binding for the `onCommand` events. The buttons appear on the page for the Model node that you associated with the Configurator Extension. To avoid confusion for end users, delete the button generated for the `updateLocationData` command; that command is issued programmatically by the Configurator Extension at run time, and clicking the button would produce an error.

5. In order to test the Interactive Location Search Configurator Extension from Configurator Developer, you must provide a custom initialization parameter named `calling_application_id`, with a value that specifies your desired calling application. You must also provide another custom initialization parameter that depends on the calling application. For Oracle Quoting and Oracle Order Management, the custom initialization parameter is `client_header` which the Extension uses to identify the customer by querying either the Order Management or Order Capture schema for the order/quote specified by the client header. See the *Oracle Configurator Implementation Guide* for details about initialization parameters.

6. Test the Configurator Extension from Configurator Developer by choosing the Test Model button, then choosing the Model Debugger, or the user interface that you generated.

Effects of the Interactive Location Search Configurator Extension

When you test the Model and click the button you created for each top-level node requiring a location, the Interactive Location Search Configurator Extension replaces the current Oracle Configurator window and points to an OA (Oracle Applications) page. The OA page determines the host application that called it, and determines the Customer ID. Then the OA page queries the TCA for all locations for that customer, and presents them so that end users can select one. When the user selects a location, the Location ID (typically a PARTY_SITE_ID) is stored as the Location ID for the runtime Component bound to the Configurator Extension.

Since this Configurator Extension redirects to another Oracle Applications Framework page and later returns to its original page, it is necessary to get a message authentication code (MAC) for the URL of the original and apply it to the URL before returning. For details, see the section on redirecting to a Framework page in the *Oracle Configurator Extensions and Interface Object Developer's Guide.*

See the section, "Location Synchronization", for important information about the synchronization of location data with Oracle Installed Base when Oracle Configurator reconfigures an instance.

If you need details on the operation of this Configurator Extension, consult the comments in its source code on MetaLink.

Modifying the Interactive Location Search Configurator Extension

Since the Interactive Location Search solution is constructed with Oracle Applications Framework in addition to Configurator Extensions, you must use the business logic written into the Java source for the controller object to modify its behavior. This requires expertise in programming for the Oracle Applications Framework.

Line Type Configurator Extension

The Line Type Configurator Extension is required for reconfiguring installed configurations of container Models.

If you do not install this extension, the Line Types for your fulfilled items will be defaulted in Oracle Order Management. The Line Type is a mandatory field on an order line. OM defaults the Line Type from the default Line Type that is set up for the order type. Line Types are also known as Actions.

If you install this extension and complete the setup steps described in this chapter, then your fulfilled orders will have Line Types in Oracle Service Fulfillment Manager. The Line Types will be those defined in this extension and the associated setup.

> **Note:** Implementing this extension assumes that you are using a user interface generated with the HTML-based version of Oracle Configurator Developer. If you want to use a DHTML user interface, you must implement a Functional Companion, as described in a previous release of this document. Functional Companions do not have all the functionality of Configurator Extensions.

The Line Type Configurator Extension determines and sets the Line Type value for the associated node and its descendants by either:

- Detecting the type of change that was made to the node being configured or reconfigured, relative to the latest valid baseline recorded in Oracle Installed Base
- Detecting a change in the state of specified Features in the Model

The Line Type is a reference to a Transaction Type in Oracle Order Management that indicates the fulfillment actions to be applied to the configured Component instances. The Configurator Extension assigns the Line Type during a configuration session when the configuration is saved or when the Summary window appears.

Setting Up Line Type Configurator Extension

The following sections describe what you must do before using the Line Type

Configurator Extension.

Setup in Order Management

Set up Oracle Order Management for use with this extension. For background and details, see the *Oracle Configurator Implementation Guide*.

1. In Oracle Order Management, define the desired Line Types, by defining Transaction Types that have a Transaction Type Code of Line.

   Transaction Types are stored in the Order Management table, ONT.OE_TRANSACTION_TYPES_TL. The name you enter in the Transaction Types window is stored as NAME. Oracle Applications stores a numeric code for the type, TRANSACTION_TYPE_ID. The NAME value is used to display the Line Type values on the Summary window of the runtime Oracle Configurator.

2. The following table shows a set of Line Types that you might define. The IDs shown are example values; the actual values of the IDs depend on the data in the particular database instance in which you define the Line Types.

Example Line Types Defined in OE_TRANSACTION_TYPES_TL

TRANSACTION_TYPE_ID	NAME
2516	MOVE
2476	ADD
2477	CHANGE
2556	DISCONTINUED
2560	SUSPEND
2570	RESUME
2600	NO CHANGE

These types are passed downstream to Oracle Service Fulfillment Manager to indicate the type of fulfillment action to perform.

1. Use the following query to see the other Transaction Types currently defined:

```
SELECT transaction_type_id, name, description
FROM ont.oe_transaction_types_tl;
```

2. Record the TRANSACTION_TYPE_ID and NAME values for your Line Types. You will use them in the task below, Setup in Oracle Configurator Developer Using User Properties, page 11-15.

Setup in Oracle Applications Database

Perform the setup below in the Oracle Configurator (CZ) schema of the Oracle Applications database when you use this Extension. For background and details, see the *Oracle Configurator Implementation Guide*.

1. Confirm that DISPLAY_SUMMARY_FULFILLMENT_ACTION is set correctly in the CZ_DB_SETTINGS table. See the chapter, Set Up Configurator and Customize the Solution, for details. The default setting should be correct.

2. After you set DISPLAY_SUMMARY_FULFILLMENT_ACTION correctly, the Summary window of the runtime Oracle Configurator displays the Line Types assigned to nodes of a BOM Model.

Setup in Oracle Configurator Developer Using User Properties

Define elements of a Model in Oracle Configurator Developer for use with this extension. For details about defining Model elements, see the *Oracle Configurator Developer User's Guide*.

1. For the Line Types that you have defined in Oracle Order Management (as shown in the example in the table Example Line Types Defined in OE_TRANSACTION_TYPES_TL, page 11-14), define sets of Properties in the Main area of the Repository. Each Line Type will require at least one set of properties, and may require more, depending on how many ways you plan to test for the Line Type. For example, a Line Type of CHANGE may need multiple tests (for attribute changes, name changes, and so on), and each test will require a set of properties. Use the settings shown in the table below. In each set of Property names, replace *n* with an integer, incrementing *n* by 1 for each set. There is no limit on the number of sets you can specify.

User Properties for Line Types

Name	Data Type	Default Value
LINETYPE_*n*_NAME	Translatable Text	(Definition is optional.)
LINETYPE_*n*_ID	Integer	(Definition is optional.)
LINETYPE_*n*_TEST	Text	(Definition is optional.)

The result should be multiple sets of Properties named in the following pattern:

```
LINETYPE_1_NAME, LINETYPE_1_ID, LINETYPE_1_TEST,
LINETYPE_2_NAME, LINETYPE_2_ID, LINETYPE_2_TEST,
LINETYPE_3_NAME, LINETYPE_3_ID, LINETYPE_3_TEST,
and so on
```

Note that you may not want to enter a default value for these Properties. If your implementation has a standard set of tests that do not vary much when applied to specific nodes, then it might be more convenient to set them as default values. But if multiple products have different Line Types and different ways of testing for them, then you should not set default values. You will be entering node-specific properties in a subsequent step.

The form of these Property names is mandatory, unless you modify the source code for this extension to search for different strings. Oracle recommends that you not make such a change.

> **Important:** In cases where the remote server for publication has TRANSACTION_TYPE_ID values that differ from those in the Configurator instance being used for development testing or staging, the published model will have incorrect Property values. For this reason Oracle recommends that you use only LINETYPE_*n*_NAME to set up the Line Type Configurator Extension. The previously documented use of LINETYPE_*n*_ID is still supported, but is deprecated. If you use both NAME and ID, their values may not match. This Extension does not check this situation, but uses the last Property that it examines.

2. For each root node in your container Model that requires Line Types, add a set of these `LINETYPE_n` Properties on that node for each applicable Line Type, with node-specific values, as described in the following steps.

   The Properties can be defined on a BOM Model, BOM Option Class, or BOM Standard Item.

3. Set the priority of the Line Type relative to the other Line Types, by selecting the set of Properties that correspond to this priority, using the integer part of the Property name to indicate the priority. The priority determines the order, at runtime, in which this extension applies tests to the associated node. The first test that passes (that is, returns a True value) causes the assignment of the corresponding Line Type to the node.

   For example:

   - LINETYPE_1_TEST is applied first.
   - If LINETYPE_1_TEST fails, then LINETYPE_2_TEST is applied.

- If LINETYPE_2_TEST passes, then the value of LINETYPE_2_NAME is assigned to the node as its Line Type.
- If LINETYPE_2_TEST fails, then LINETYPE_3_TEST is applied, and so on.

4. For the LINETYPE_*n*_NAME (or LINETYPE_*n*_ID) member of the Property set, enter the instance-independent NAME (or instance-specific TRANSACTION_TYPE_ID) for the Line Type, as shown in the table, Example Line Types Defined in OE_TRANSACTION_TYPES_TL, page 11-14.

   For example, for the Line Type called ADD, enter `ADD` as the value of the Property LINETYPE_1_NAME (or enter `2476` as the value of the Property LINETYPE_1_ID).

   The value you enter for LINETYPE_*n*_NAME is of type Translatable Text. You must ensure that this name is unique for the translated language in which you enter it, and that it corresponds exactly to the translated value of the NAME defined in Order Management table ONT.OE_TRANSACTION_TYPES_TL.

5. For the LINETYPE_*n*_TEST member of the Property set, enter a token string which indicates the CIO test that determines whether the specified type of change occurred that corresponds to the Line Type. The token string is composed of the string `&BaseNode.` combined with one of the predefined CIO test methods listed in the table, Methods for Identifying Changes Against the Baseline, page 11-30, later in this chapter. (If you wish to use the method `isIBNodeOrDescendantChanged()`, you must customize the code of this extension.)

   For example, for the Line Type called ADD, you would probably enter the following token string:

   ```
   &BaseNode.isAddChanged()
   ```

   If you use the method `isTargetChanged()`, then you must use the token string syntax `&BaseNode.Connector`*n*`.isTargetChanged()`, for each Connector, since the target change is on the Connector, not on the Base Node.

   You must determine which test corresponds to your own definition of the meaning of a particular Line Type. If the CIO tests listed in the table, Methods for Identifying Changes Against the Baseline, do not capture some of your Line Types, then see the section, Setup in Oracle Configurator Developer Using Features, page 11-21.

6. When you have finished defining your Property sets, the results should resemble the contents of the table Line Type Definitions Compared to User Properties for Line Types, page 11-18, which combines the Line Types shown in the table Example Line Types Defined in OE_TRANSACTION_TYPES_TL, page 11-14 with the Property definition sets that correspond to each of them. The Line Type priorities shown for LINETYPE_*n* are for demonstration only. You must assign your own priorities.

Line Type Definitions Compared to User Properties for Line Types

TRANSACTION_TYPE_ID	NAME	*n*	LINETYPE_*n*_NAME	LINETYPE_*n*_ID	LINETYPE_*n*_TEST
2476	ADD	1	ADD	2476	`&BaseNode.isAddChanged()`
2556	DISCONTINUED	2	DISCONTINUED	2556	`&BaseNode.isDeleteChanged()`
2477	CHANGE	3	CHANGE	2477	`&BaseNode.isAttributeChanged()`
2516	MOVE	4	MOVE	2516	`&BaseNode.isLocationChanged()`

Setup in Oracle Configurator Developer for Missing Nodes

Missing nodes correspond to situations where Oracle Installed Base contains a configuration having elements that are no longer present in the published model currently being used for orders.

Example: a model contains a Value Added Service named "Free Internet". This service was removed from the configuration model's definition, but there are item instances in Installed Base where it may be selected. When such an instance is reconfigured, the service cannot be fully restored, because this service is no longer a node in the configuration model definition. If a Line Type is to be applied to the service, it has to be done without applying any tests on the node, because the node does not exist in the current reconfiguration session.

During reconfiguration, the Line Type Configurator Extension can identify the BOM nodes that are no longer present in the item instance that is being reconfigured. This section describes how to assign a Line Type to those nodes.

See Setup in Oracle Configurator Developer Using User Properties, page 11-15 for important background to this setup.

1. Define a set of Properties in the Main area of the Repository according to the following table.

User Properties for Line Types on Missing Nodes

Name	Data Type	Default Value
LINETYPE_MISSING_NAME	Translatable Text	(Definition is optional.)
LINETYPE_MISSING	Integer	(Do not define one.)

2. For each trackable root BOM Model that requires Line Types, add either of these LINETYPE_MISSING Properties to that node. Add the Property only once per each trackable root, which must be a BOM Model (not a BOM Option Class or BOM Standard Item).

3. Assign a value to the Property to indicate the appropriate Line Type, as described below.

   In the Property LINETYPE_MISSING_NAME (or LINETYPE_MISSING ID), enter the instance-independent NAME (or instance-specific TRANSACTION_TYPE_ID) for the "missing" Line Type as defined in Order Management. The value indicates the Line Type that you want to assign if the node is found to be missing during reconfiguration, relative to the instance restored from Installed Base.

   Example values are shown in the table Example Line Types Defined in OE_TRANSACTION_TYPES_TL, page 11-14. For example, if the defined Line Type is called DISCONTINUED, enter `DISCONTINUED` as the value of the Property LINETYPE_MISSING_NAME. If you are using Property IDs instead of names, enter the instance-specific TRANSACTION_TYPE_ID (such as 2556) as the value of the Property LINETYPE_MISSING). While it is possible to enter different Property values for different trackable root nodes, this is probably not useful and is not recommended.

   The value you enter for LINETYPE_MISSING_NAME is of type Translatable Text. You must ensure that this name is unique for the translated language in which you enter it, and that it corresponds exactly to the translated value of the NAME defined in Order Management table ONT.OE_TRANSACTION_TYPES_TL.

Setup in Oracle Configurator Developer for Unchanged Nodes

During reconfiguration, the Line Type Configurator Extension can identify BOM nodes that have no actual changes made to them, and so are not considered "changed" when the item instance is being reconfigured. This section describes how to assign a Line Type to those nodes.

Although some implementers want these lines to appear in the quote or order resulting

from a reconfiguration, Oracle recommends against this practice, for performance reasons.

See Setup in Oracle Configurator Developer Using User Properties, page 11-15 for important background to this setup.

1. Define a set of Properties in the Main area of the Repository according to the following table.

User Properties for Line Types on Unchanged Nodes

Name	Data Type	Default Value
LINETYPE_UNCHANGED_NAME	Translatable Text	(Definition is optional.)
LINETYPE_UNCHANGED	Integer	(Do not define one.)

2. For each trackable root BOM Model that requires Line Types, add either of these LINETYPE_UNCHANGED Properties to that node. Add the Property only once per each trackable root, which must be a BOM Model (not a BOM Option Class or BOM Standard Item).

3. Assign a value to the Property to indicate the appropriate Line Type, as described below.

   In the Property LINETYPE_UNCHANGED_NAME (or LINETYPE_UNCHANGED ID), enter the instance-independent NAME (or instance-specific TRANSACTION_TYPE_ID) for the "unchanged" Line Type as defined in Order Management. The value indicates the Line Type that you want to assign if the node is found to be unchanged during reconfiguration.

   This value is used to ensure that merely assigning a Line Type does not trigger a "change" Line Type on parent items. (By default, Line Type assignment to any child implies a change to that child, resulting in parent items returning true for the `isChildChanged()` method.)

   Example values are shown in the table Example Line Types Defined in OE_TRANSACTION_TYPES_TL, page 11-14. For example, if the defined Line Type is called NO CHANGE, enter `NO CHANGE` as the value of the Property LINETYPE_UNCHANGED_NAME. If you are using Property IDs instead of names, enter the instance-specific TRANSACTION_TYPE_ID (such as 2600) as the value of the Property LINETYPE_UNCHANGED). While it is possible to enter different Property values for different trackable root nodes, this is probably not useful and is not recommended.

   The value you enter for LINETYPE_UNCHANGED_NAME is of type Translatable

Text. You must ensure that this name is unique for the translated language in which you enter it, and that it corresponds exactly to the translated value of the NAME defined in Order Management table ONT.OE_TRANSACTION_TYPES_TL.

Setup in Oracle Configurator Developer Using Features

If the CIO tests listed in the table Methods for Identifying Changes Against the Baseline, page 11-30, later in this chapter, do not capture the meaning of some of your Line Types, then you can Model those Line Types as nodes whose state can be tested as being True or False. You can Model a Line Type test as either the True state of a Boolean Feature or as the selected Option of an Option Feature.

1. Determine which of your Line Types (such as those listed in Example Line Types Defined in OE_TRANSACTION_TYPES_TL, page 11-14) are not captured by the CIO tests listed in the table Methods for Identifying Changes Against the Baseline, page 11-30.

2. Define a Feature whose state, when it is evaluated as True, corresponds to that Line Type. You might choose to use either an Option Feature or a Boolean Feature.

   - Define an Option Feature whose options correspond to the names of those Line Types.

     For example:

     - Define an Option Feature named `Use Cases`
     - For the Line Type called SUSPEND, define an option of `Use Cases` named `Suspend`
     - For the Line Type called RESUME, define an option of `Use Cases` named `Resume`

   - Define Boolean Features whose True states correspond to the names of those Line Types.

     For example:

     - For the Line Type called SUSPEND, define a Boolean Feature named `Suspended?`
     - For the Line Type called RESUME, define a Boolean Feature named `Resumed?`

3. Follow steps 1 through 4 under the section Setup in Oracle Configurator Developer Using Properties, page 11-15.

4. For the LINETYPE_*n*_TEST member of the Property set, enter the relative node path from the nearest parent BOM Model to the Option or Boolean Feature that

corresponds to the Line Type.

For example, for the Line Type called SUSPEND, you would enter the following path: `Use Cases.Suspend`

5. When you have finished defining your Property sets, using both the Property-based and Feature-based methods, the results might resemble the contents of the table More Line Type Definitions Compared to User Properties for Line Types, page 11-22, which combines the Line Types shown earlier with the Property definition sets that correspond to each of them. The table below is similar to the earlier table Line Type Definitions Compared to User Properties for Line Types, page 11-18, with the addition of rows showing Feature-based definitions for the Line Types SUSPEND and RESUME. For the sake of demonstration, the table shows both an Option Feature-based value and a Boolean Feature-based value for LINETYPE_*n*_TEST. This is not necessarily a recommended practice.

   The Line Type priorities shown for LINETYPE_*n* are for demonstration only. You must assign your own priorities.

More Line Type Definitions Compared to User Properties for Line Types

TRANSACTION_TYPE_ID	NAME	*n*	LINETYPE_*n*_NAME	LINETYPE_*n*_ID	LINETYPE_*n*_TEST
2476	ADD	1	ADD	2476	`&BaseNode.isAddChanged()`
2556	DISCONTINUED	2	DISCONTINUED	2556	`&BaseNode.isDeleteChanged()`
2477	CHANGE	3	CHANGE	2477	`&BaseNode.isValueChanged()`
2516	MOVE	4	MOVE	2516	`&BaseNode.isLocationChanged()`
2560	SUSPEND	5	SUSPEND	2560	`Use Cases.Suspend`
2570	RESUME	6	RESUME	2570	`Resumed?`

Setup in Oracle Configurator Developer Using Indirection

The setup described elsewhere in this section assumes that some of the items in a trackable Component instance share the same data that this Configurator Extension uses to assign the Line Type (that is, the Line Type's ID, test, and priority).

It is very possible that your own configuration Model requires different IDs, tests, or priorities for different trackable items. The different items might be other trackable root Components or trackable descendants of the Component you have already defined Line Types for. You could accomplish this distinction by defining User Properties on all the affected items, but this course might require many Property definitions.

To remedy this situation, this extension allows you to define Line Type data for a node by specifying only the path to a node that already has the required Property definitions. This method is called *indirection*, since the Property definitions for a node are obtained indirectly from another node.

1. In the Main area of the Repository, define a single Text Property, named LINETYPE_CHECKLIST, with no default value.

2. Determine which of your trackable nodes require Property definitions for Line Types that differ from the set that you have defined on the originally selected node.

3. In one of those differing nodes, define the Property sets for Line Types, as described in the section Setup in Oracle Configurator Developer Using Properties, page 11-15.

   If the differing node is a trackable descendant of the originally selected node, then at runtime it reuses the Line Type-related User Properties of that ancestor node. Consequently, you do not have to define a full set of Property sets on the differing child node; you only have to define the Property sets with values that differ from the original node's. Any Property definitions that are defined locally on a node override the reused definitions.

4. In one of the other differing nodes, add the User Property LINETYPE_CHECKLIST that you defined.

5. For the value of the Property LINETYPE_CHECKLIST, enter the relative node path from the nearest parent BOM Model to the differing node that carries the Properties that you defined in step 3.

6. Expanding on the explanation under step 3, the Properties of the node specified by LINETYPE_CHECKLIST are reused. If you need to further redefine the Properties for one of the differing nodes, you only have to define the Property sets with values that differ from the checklist node. These local values override the inherited ones.

7. Repeat steps 4 through 6 for all of the other differing nodes in your Model.

Define the Configurator Extension Rule

Important: Be sure to check the MetaLink Note for this Configurator Extension, as listed in Customizable Source Code for TSO Configurator

Extensions, page 11-5, to see whether there have been updates to the source code or rule binding instructions.

You need to define a Configurator Extension Rule in Oracle Configurator Developer for use with this extension. For details about defining Configurator Extension Rules, see the *Oracle Configurator Developer User's Guide*.

1. If you are customizing the behavior of this Configurator Extension, then see Using Customizable Source Code for TSO Configurator Extensions, page 11-4. If you are using the installed classes and not customizing, see Using Installed Classes for TSO Configurator Extensions, page 11-4.

2. Define a Configurator Extension Rule with the options listed below:
   - **Model Node:** Choose the node of your Model whose instances need their Line Types updated. The update applies to all descendants of the Component.
   - **Java Class:** Specify `oracle.apps.cz.cx.tso.AssignItemLineTypeCX`.
   - **Java Class Instantiation:** Choose `With Model Node Instance`.

3. Create an **event binding** for the above Configurator Extension Rule, with the options listed below:
   - **Event:** Choose `postInstanceLoad`.
   - **Event Scope:** Choose `Base Node Subtree`.
   - **Method:** Choose `initializeItemLineTypeMapping(oracle.apps.cz.cio.IRuntimeNode Arg1)`.

4. Create an **argument binding** for the above event binding, with the options listed below:
   - **Argument Name:** Accept `node` or `Arg1`.
   - **Argument Type:** Accept `oracle.apps.cz.cio.IRuntimeNode`.
   - **Argument Specification:** Choose `Event Parameter`.
   - **Binding:** Choose `newNode`.

5. Create a second **event binding** for the Configurator Extension Rule, with the options listed below:
   - **Event:** Choose `onConfigLineType`.

- **Event Scope:** Choose `Global`.
- **Method:** Choose `assignItemLineType(oracle.apps.cz.cio.IRuntimeNode Arg1)`.

6. Create an **argument binding** for the above event binding, with the options listed below:
   - **Argument Name:** Accept `trackableRoot` or `Arg1`.
   - **Argument Type:** Accept `oracle.apps.cz.cio.IRuntimeNode`.
   - **Argument Specification:** Choose `System Parameter`.
   - **Binding:** Choose `Base Node of Rule`.
7. Generate logic for your Model, in order to reflect the addition of the Configurator Extension Rule.
8. Create or refresh a user interface for your Model.
9. Test the Configurator Extension from Configurator Developer by choosing the Test Model button, then choosing the Model Debugger, or the user interface that you generated.

Effects of Line Type Configurator Extension

The method `initializeItemLineTypeMapping()` is bound to the event `postInstanceLoad` so it is triggered immediately after a Component instance or other associated node is created, or brought into the configuration. This method validates that all the Line Type-related Property data is correct, and then initializes the mapping structures that will cache this data for use by the `assignItemLineType()` method.

The method `assignItemLineType()` is bound to the event `onConfigLineType`, which is triggered whenever the Summary window appears and displays the Line Type values. Choosing , at runtime, to display only the changes relative to Installed Base also triggers this event. The `onConfigLineType` event is also triggered when the current configuration is saved, such as when the user ends a configuration session. This method proceeds recursively through each BOM entity in the configuration, assigning the Line Types, based on the Property data cached by the method, `initializeItemLineTypeMapping()`. Line Types can be assigned to both trackable and non-trackable BOM nodes.

Because of the interdependency of the two methods, it is essential to perform the event binding for both of them.

If you need details on the operation of this Configurator Extension, consult the comments in its source code on MetaLink.

Modifying Line Type Configurator Extension

If you modify your Model in accordance with the steps in this section, then you should not need to modify the Line Type Configurator Extension.

IBAttribute Configurator Extension

The IBAttribute Configurator Extension allows end users to collect attributes of configured Components. This extension is recommended, but not required for implementing the reconfiguring of installed configurations of container Models.

> **Note:** Implementing this extension assumes that you are using a user interface generated with the HTML-based version of Oracle Configurator Developer. If you want to use a DHTML user interface, you must implement a Functional Companion, as described in a previous release of this document. Functional Companions do not have all the functionality of Configurator Extensions.

This extension enables an end user of the runtime Oracle Configurator to collect the values of attribute Features that have been set on the trackable children of a runtime Component instance. When the configuration is saved, the configuration attribute values are automatically written to the pending transactions for Oracle Installed Base.

Setting Up IBAttribute Configurator Extension

The following sections describe what you must do before using the IBAttribute Configurator Extension.

General Setup

Set up elements of the Oracle Applications database for use with this extension. For background and details, see the *Oracle Configurator Implementation Guide*. For trackable items, map extended attributes in Oracle Installed Base to configuration attribute Features and Properties in Configurator Developer.

For more information, see the chapter, Set Up Configurator and Customize the Solution.

Define the Configurator Extension Rule

> **Important:** Be sure to check the MetaLink Note for this Configurator Extension, as listed in Customizable Source Code for TSO Configurator

Extensions, page 11-5, to see whether there have been updates to the source code or rule binding instructions.

You need to define a Configurator Extension Rule in Oracle Configurator Developer for use with this extension. For details about defining Configurator Extension Rules, see the *Oracle Configurator Developer User's Guide*.

1. If you are customizing the behavior of this Configurator Extension, then see Using Customizable Source Code for TSO Configurator Extensions, page 11-4. If you are using the installed classes and not customizing, see Using Installed Classes for TSO Configurator Extensions, page 11-4.

2. Define a Configurator Extension Rule with the options listed below:
   - **Model Node:** Choose the node of your Model whose instances need their attributes collected. The Configurator Extension searches all descendants of the Component for attributes.
   - **Java Class:** Specify `oracle.apps.cz.cx.tso.CZIBAttributeCX`.
   - **Java Class Instantiation:** Choose `With Model Node Instance`.

3. Create an **event binding** for the above Configurator Extension Rule, with the options listed below:
   - **Event:** Choose `onInstanceLoad`.
   - **Event Scope:** Choose `Base Node Subtree`.
   - **Method:** Choose `onLoad(oracle.apps.cz.cio.IRuntimeNode Arg1)`.

4. Create an **argument binding** for the above event binding, with the options listed below:
   - **Argument Name:** Accept `node` or `Arg1`.
   - **Argument Type:** Accept `oracle.apps.cz.cio.IRuntimeNode`.
   - **Argument Specification:** Choose `Event Parameter`.
   - **Binding:** Choose `newNode`.

5. Generate logic for your Model, in order to reflect the addition of the Configurator Extension Rule.

6. Create or refresh a user interface for your Model.

7. Test the Configurator Extension from Configurator Developer by choosing the Test Model button, then choosing the Model Debugger, or the user interface that you generated.

Effects of the IBAttribute Configurator Extension

The IBAttribute Configurator Extension runs whenever a trackable instance is loaded. For each trackable BOM child of the instance, the Configurator Extension:

- Collects configuration attribute information from the Properties.
- Finds the attribute Feature.
- Associates the attribute with the RuntimeNode.
- Propagates the attribute to the node's children (including instantiable Components) based on the propagation mode. The IBAttribute Configurator Extension associates attributes with a RuntimeNode by using the public class `Attribute`, which is part of the CIO.

Finally, the attribute values are saved to the CZ_CONFIG_EXT_ATTRIBUTES table when the configuration is saved. The CZ_CONFIG_EXT_ATTRIBUTES table is a temporary holder for the attribute values while the attribute details are pending for acceptance in Installed Base. At this point, the attributes become part of the transaction details in the transaction that is pending for acceptance into Oracle Installed Base. When the transaction is accepted (by installation through Service Fulfillment Manager), the configuration attribute values become part of the installed instance.

See the section Attribute Synchronization, page 11-36, for important information about the synchronization of attribute data with Oracle Installed Base when Oracle Configurator reconfigures an instance.

If you need details on the operation of this Configurator Extension, consult the comments in its source code on MetaLink.

Modifying the IBAttribute Configurator Extension

For information on using the classes and methods of the Configuration Interface Object (CIO), see the *Oracle Configurator Extensions and Interface Object Developer's Guide*.

If you modify your Model in accordance with the steps in the section Setting Up IBAttribute Configurator Extension, page 11-26, then you do not need to modify the IBAttribute Configurator Extension.

When modifying your Model, you must follow the conventions for naming Properties described in the *Oracle Configurator Methodologies* chapter on configuration attributes. Otherwise, the IBAttribute Configurator Extension will not be able to collect values for

the attribute Features. To use different Property names, you must modify the character strings used to match Property names, so that they match your own Property names. The following example shows typical matching strings.

Example: Strings for Property Names in the Configurator Extension

```
...
  private void getAttributes(RuntimeNode node) {
...
        if (prop.getName().startsWith("ATTR_")) {
...
        int beginningIndex = new String("ATTR_"). length();
    StringTokenizer tokens = new
StringTokenizer(name.substring(beginningIndex), "_", false);
...
        if (name.endsWith("PATH")) {
...
    } else if (name.endsWith("MODE")) {
...
    } else if (name.endsWith("NAME")) {
...
    } else if (name.endsWith("GROUP")) {
...
      if (prop.getName().startsWith("ATTR_NAME")) {
...
```

Using the Oracle Configuration Interface Object (CIO)

This section contains information about using the Oracle Configuration Interface Object (CIO). For additional information about using this API, see the *Oracle Configurator Extensions and Interface Object Developer's Guide*.

Reconfigure Installed Configurations

In addition to single-instance configurations, Oracle Configurator can create and maintain configurations that consist of multiple instances of trackable configurable Components. That is, the instances can be joined into a network by creating connections. For more information about instance configurations, see the *Oracle Configurator Extensions and Interface Object Developer's Guide*.

Each Component instance has a unique Instance Header identifier, stored as CZ_CONFIG_ITEMS.INSTANCE_HDR_ID. You can save, restore, revise, and save again a Component instance. Each saved instance revision has a non-unique Instance Revision Number identifier, stored as CZ_CONFIG_ITEMS.INSTANCE_REV_NBR. The unique combination of Header ID and Revision Number identifies an instance record. For more information on identifying configurations, see the *Oracle Configurator Implementation Guide*.

Configurations can include multiple Component instances, and you can configure the same Component instance in multiple configuration sessions.

In Oracle Installed Base, you can track the baseline and revised Component instances in a configuration. For information about how to determine differences between the baseline for a Component in Oracle Installed Base and the revised instance of that

Component, see the section, "Identify Changes to a Configuration", below.

For information on creating configuration Models that support reconfiguration, see the chapter, Set Up Configurator and Customize the Solution.

Identify Changes to a Configuration

You can use the set of methods appearing in the table, Methods for Identifying Changes Against the Baseline, below, to identify the types of changes that occur during a configuration session between the baseline revision of the trackable Component instance (as installed in Oracle Installed Base when the configuration session began) and the current Component instance (which is being reconfigured). The methods appearing in the table are all members of the `IRuntimeNode` interface.

Methods for Identifying Changes Against the Baseline

Method	Returns True If...
`isIBNodeOrDescendantChanged()`	Any of the other methods listed here return True.
`isAddChanged()`	This node has been added in this session, meaning that it is not present in the installed revision.
`isDeleteChanged()`	This node has been deleted in this session and was present in the installed revision.
`isValueChanged()`	This node's value has changed from that of the installed revision.
`isTargetChanged()`	This Connector's target has changed from that of the installed revision.
`isConnectivityChanged()`	The list of connected-to relationships for the currently configured revision of a trackable component instance has changed from that of the installed revision. This method also returns True on the parent of a Connector if the Connector's target has changed from that of the installed revision.

Method	Returns True If...
`isAttributeChanged()`	Any of this node's configuration attributes have changed from those of the latest installed revision. A transient attribute is considered changed only if there is a change to value of the attribute Feature.
`isLocationChanged()`	This node's Location or Location Type Code has changed from that of the installed revision.
`isNameChanged()`	This instance's name has changed from that of the installed revision.
`isChildChanged()`	Any descendant of this node has changed from that of the installed revision.

Following is a fragmentary example of how to use one of these methods.

```
...
public boolean wasAdded(RuntimeNode node) {
    boolean added = node.isAddChanged();
    return added;
}
...
```

For additional explanation, see the section, "Specify the Line Type", earlier in this chapter.

Programmatically Reconfigure Installed Instances

You can reconfigure installed instances directly through the CIO. To ensure that you select all the trackable instances that you want to reconfigure, you need to identify the set of instances and add them to a `ConfigParameters` object before calling `CIO.startConfiguration()`.

Following are suggestions for determining the set of instances that you want to reconfigure.

- If you know the order number for the fulfilled instances, then you can query for the order number as the value of CSI_ITEM_INSTANCES.LAST_OE_ORDER_LINE_ID. The instances in Installed Base for that order are identified by CSI_ITEM_INSTANCES.CONFIG_INST_HDR_ID.

- If you know the Session Header ID (CZ_CONFIG_DETAILS_V.CONFIG_HDR_ID) and Session Revision Number (CZ_CONFIG_DETAILS_V.CONFIG_REV_NBR) of the order that was fulfilled, then you can use the following query to find the instances in that order.

  Query for Instances

```
SELECT config_inst_hdr_id, config_inst_rev_num
  FROM apps.csi_item_instances ib, apps.cz_config_details_v cz
  WHERE ib.config_inst_hdr_id = cz.instance_hdr_id
  AND cz.component_instance_type = 'I'
  AND cz.config_hdr_id = session_header_id
  AND cz.config_rev_nbr = session_revision_number
  AND cz.config_item_id = ib.config_inst_item_id
  AND ib.active_end_date IS NULL;
```

Once you have identified the desired instances, use `ConfigParameters.addInstalledInstance()` to add them to the `ConfigParameters` object used to start the reconfiguration, then call `CIO.startConfiguration()` to create a configuration containing the instances.

The following example, Reconfiguring Installed Instances, page 11-33, demonstrates this technique. The code assumes that you have created a CIO object and a database context, and have identified the desired model. To identify the model, a hard-coded hypothetical model ID is shown. You might also define the custom method shown, `getPublication(itemId, orgId)`. You pass the `Collection` of desired instances to the custom method `startReconfigFlow()`.

Reconfiguring Installed Instances

```
import com.sun.java.util.collections.Collection;
import com.sun.java.util.collections.Iterator;

import oracle.apps.cz.cio.BomExplosionException;
import oracle.apps.cz.cio.CIO;
import oracle.apps.cz.cio.ConfigParameters;
import oracle.apps.cz.cio.Configuration;
import oracle.apps.cz.cio.LogicalException;
import oracle.apps.cz.cio.ModelLookupException;
import oracle.apps.cz.dio.ui.NoSuchUiDefException;
import oracle.apps.cz.utilities.EffectivityUsageException;
import oracle.apps.fnd.common.Context;


public class Reconfiguration {
  public void startReconfigFlow(CIO cio, Context ctx, int itemId, int
orgId, Collection instances)
      throws LogicalException, ModelLookupException,
EffectivityUsageException, BomExplosionException, NoSuchUiDefException {

    ConfigParameters params = null;
    int modelId = 1000; // or define custom method:
getPublication(itemId, orgId);
    params = new ConfigParameters( modelId );
    params.setValidationContext("INSTALLED");
    if(instances != null) {
        for(Iterator itr =  instances.iterator(); itr.hasNext(); ) {
            Long insHdr =  (Long) itr.next();
            params.addInstalledInstance( insHdr );
        }
    }
    Configuration config  = cio.startConfiguration(params, ctx);
    // perform reconfiguration of instances here
    config.completeBeforeSave();
    config.saveNew();
  }
}
```

About Discontinued Items

An item node is considered discontinued if it was previously installed and, during a configuration session, it is either:

- *deselected*: the end user deselects the node, thus setting its state to unknown or false
- *deleted*: the end user removes the node (for example, by clicking the Delete button on an instance containing the node)

Be aware that for downstream applications there is no effective difference between these states.

> **Note:** Oracle recommends not deleting or deselecting instances at runtime, if possible. Instead, we suggest using a Boolean Feature that indicates the Line Type. This approach allows the end user to access the

instance and supply additional attribute data, such as the reason for the discontinuation. The Line Type may be used in SFM to expire the instance so that it is no longer accessible for reconfiguration. Using this approach makes it much less likely that you need to worry about the concerns in the section Access Discontinued Items, page 11-34.

Access Discontinued Items

To access both current and discontinued items in a TSO solution, you must use the following alternate signatures for these methods:

- `RuntimeNode.getChildByName(String name, int type)`
- `RuntimeNode.getChildByID(int ID, int type)`
- `RuntimeNode.getChildByPersistentID(int ID, int type)`

Each signature includes a second argument, type. The possible values for type are:

- `RuntimeNode.CURRENT_OR_DISCONTINUED_CHILD`
- `RuntimeNode.CURRENT_CHILD`
- `RuntimeNode.DISCONTINUED_CHILD`

The default type is `CURRENT_CHILD`.

If a configured instance of your Model might contain discontinued nodes, then you should also call `IRuntimeNode.isDiscontinued()` as a condition of working with a node, such as by calling `getState()`, `getCount()`, or `isSelected()`. If a node has been discontinued by deselection, but not by deletion, then calling a method on it does not raise a NodeDeletedException.

You should also be familiar with the list of problems that can occur when evaluating runtime conditions provided in the *Oracle Configurator Developer User's Guide*, since some of those problems may resemble issues with accessing discontinued nodes.

Access Instances

To access trackable Component instances in a TSO solution, use one of the following methods:

- Use `Component.getInstanceNumberLong()` to get the Instance Header ID of a trackable Component instance. If the Component instance is not trackable, this method returns the Instance Number. If the Component is a mandatory Component (Minimum Instances and Maximum Instances are both 1), this method returns 1.
- Use `ComponentSet.getChildByInstanceNumber(int instNum)` to get a trackable Component instance by passing an Instance Header ID.

- If you have set the name for a Component instance--by using `Component.setInstanceName(String newName)`--then you can get that instance by the name you set, by using `ComponentSet.getChildByName(String userSetName)` on the `ComponentSet` that includes the instance.

- In order to access both current and discontinued Component instances, use the following alternate signature: `ComponentSet.getChildByInstanceNumber(int instNum, int type)`. For type, specify one of the values listed above (such as CURRENT_CHILD).

Report on Link Items

If the link flag on a node is set by Oracle Configurator, then this marks it as a link. Links are trackable, but do not have a location. See the chapter, Set Up Configurator and Customize the Solution, for database details.

The CIO reports as unsatisfied any Model instance that is:

- A trackable root

- Does not have a location assigned

The following methods report whether a node is a link:

- `BomInstance.isIBLinkItem()`

- `ITrackableInstance.isIBLinkItem()`

Event-Driven Behavior for Configuration

The use of events to trigger Configurator Extensions is a notable feature of Oracle Configurator. The TSO solution does not introduce any new events or restrict the use of existing events. However, there are some events with particular significance for TSO. The following table lists these events. For details on Configurator Extension events, see the *Oracle Configurator Developer User's Guide*.

Events Specially Related to TSO

Event	Description	Comments
`onInstanceLoad`	Event dispatched when a Component instance or other associated node is created, or brought into the configuration.	This event is dispatched before synchronization of attribute data with Oracle Installed Base. The primary purpose of this event is for synchronizing attributes. This event is specific to the TSO solution.
`postInstanceLoad`	Event dispatched immediately after a Component instance or other associated node is created, or brought into the configuration.	This event is dispatched after synchronization of attribute data with Oracle Installed Base. This event is the best one to use for inspecting the instance before presenting it in the UI to end users. This event is not specific to the TSO solution.

Attribute Synchronization

During the fulfillment phase of the TSO flow, end users can change the attributes of instances of trackable items. These changes are saved in Oracle Installed Base, but are not kept as data in the saved configuration results. When Oracle Configurator reconfigures an instance, it synchronizes the attribute data for the item's saved configuration with the latest attribute data from Oracle Installed Base. This synchronization occurs in the following order relative to events listed in the above table.

1. The `onInstanceLoad` event is dispatched.
2. Configurator synchronizes the attribute data in the saved configuration of an instance with the latest attribute data from Oracle Installed Base (except for transient attributes).
3. The `postInstanceLoad` event is dispatched.

Note: If you create a Configurator Extension that depends on attribute data, ensure that you bind it to the proper event. If you are creating attribute data before fulfillment,

choose `onInstanceLoad`. If you are restoring attribute data after fulfillment, choose `postInstanceLoad`.

See the section, "Use Configuration Attributes with Installed Base (IB)" earlier in this chapter, for background on how attribute values are assigned.

Transient attribute values are not restored during attribute synchronization. See the section Transient Attributes, page 10-12 for background.

Setting Extended Attribute Values

The CIO allows you to directly set the value of an extended attribute. To do so, use `RuntimeNode.setAttributeValue(String name, String group, String value)`, specifying the name and group of the attribute, and the value to set for it. The `group` argument can be null.

Instance Name Synchronization

During reconfiguration, Oracle Configurator synchronizes the instance name for the item's configuration with the latest instance name from Oracle Installed Base. If you have changed the instance name during fulfillment, the name is restored when you reconfigure the instance in the runtime Configurator. The synchronization of instance names has the same relation to Configurator Extension events as the synchronization of attribute data described in the section, "Attribute Synchronization", above.

Location Synchronization

During reconfiguration, Oracle Configurator synchronizes the location for the item's configuration with the latest location from Oracle Installed Base. If you have changed the location during fulfillment, the location is restored when you reconfigure the instance in the runtime Configurator. The synchronization of location has the same relation to Configurator Extension events as the synchronization of attribute data described in the section, "Attribute Synchronization", above.

Accessing Tangible Items

A BOM node is tangible if it is a shippable, inventory transactable, serializable, non-ATO standard item, in a Network Configuration.

- `BomNode.isTangible()` returns True if this BOM node is tangible.
- `BomNode.isReturned()` returns True if this tangible node was returned in Installed Base.
-

For details on tangible items, see Tangible Items, page 10-14.

Requiring Text Input

You can make input for a Text Feature required by calling `TextFeature.setRequired()`, as described in the *Oracle Configurator Extensions and Interface Object Developer's Guide*. However, when using this technique in the TSO flow, do not call `TextFeature.setRequired()` on passive item instances. Doing so will produce a runtime error when the current transaction is committed or rolled back. From the standpoint of the CIO, a passive item instance is one that returns False when tested with `RuntimeNode.isEditable()`. For background on passive item instances, see Partial Network Reconfiguration and Validation, page 3-4. You should only call `TextFeature.setRequired()` in a Configurator Extension that is bound to the event `postInstanceEditable`.

12

Set Up Oracle Contact Center

This chapter covers the following topics:

- Overview of Set Up Oracle Contact Center Chapter
- Actions Architecture Overview
- Set Up Additional Actions

Overview of Set Up Oracle Contact Center Chapter

This chapter describes setup tasks that you must perform in Oracle Contact Center that are specific to the Oracle Telecommunications Service Ordering (TSO) solution.

Before You Begin

Before implementing Oracle Contact Center, you must set up the core applications for the Oracle TSO solution. For details, refer to the chapter, Implementing the Oracle TSO Solution, page 4-1.

Setup Checklist

Following is the setup checklist for implementing TSO-specific functionality for Oracle Contact Center.

Oracle Contact Center-TSO Setup Checklist

Setup Step	Required/Optional
Implement Oracle Contact Center	Required
Set Up Additional Actions	Optional

For more information on setting up Oracle Contact Center, refer to the *Oracle TeleService Implementation Guide*.

> **Note:** Two call reasons -- DisconnectService and ReconfigureService -- are supplied for the reconfigure and disconnect Oracle Telecommunication Services Ordering (TSO) business flows. Refer to the Enabling Telephony in the Contact Center chapter of the *Oracle TeleService Implementation Guide* for more information.

Actions Architecture Overview

The Install Base and Orders tabs of the Contact Center include actions agents can use to speed up tasks such as the creation of service requests for an install base item or the creation of a new item instance. Implementers can use the Oracle Applications Form Personalization to create new actions of your own or add code using the CUSTOM.pll stub library. Implementers can create different sets of actions for different classes of users (which menu is available for which user is determined by setting a system profile option). Implementers have more flexibility when you use the CUSTOM.pll stub library because it has complete access to all PL/SQL and SQL. However, Form Personalization can handle the vast majority of your changes. Form Personalization and CUSTOM library changes can co-exist. Whenever an event is fired, Form Personalization processes them first, then passes them to the CUSTOM library. Form Personalization makes it possible for implementers to declaratively alter the behavior of Oracle Forms-based windows, including changing properties, executing built-ins, displaying messages, and adding menu entries. For each function (a form running in a particular context based on parameters passed to it), implementers can specify one or more rules. Each rule consists of an event, an optional condition and its scope, and one or more actions to perform. To use Form Personalization, implementers must be familiar with Oracle Forms, the PL/SQL programming language, and the *Oracle Applications Developer's Guide*. Additionally, any changes made can interfere with the base code of a form (the code that Oracle ships). For more information on how to use Form Personalization, see the documents contained in Note 279034.1 on Oracle*MetaLink*.

> **Note:** The Actions menu in the Contact Center Orders tab can be implemented as a button or a drop-down list, depending on the value of the Oracle Order Management lookup, OM_BUTTON_POPLIST; the lookup has two values - Button or Poplist (drop-down list).

For more information on the Actions that can be performed in Oracle Contact Center, refer to the chapter, Using the Oracle TSO Solution, page 16-1.

Set Up Additional Actions

The Install Base and Orders tabs within Oracle Contact Center contain seeded actions

(described in the chapter, Using the Oracle TSO Solution, page 16-1). Depending upon business requirements, implementers may wish to add additional actions specific to TSO. The following sections describe how to set up new actions for the Install Base and Orders tabs.

Prerequisite: Familiarity with Oracle Applications architecture and Oracle Applications Form Personalization.

To set up actions for the Install Base tab:

1. Using Application Developer responsibility, create TSO functions (actions) for the Install Base tab and either add the functions to the seeded menu, CSI_CC_ACTIONS_SUB_MENU (which has the user menu name of Install Base Tab Actions sub menu) or create a new menu and add the new functions to it. Use Form Personalization or the CUSTOM.pll to write the code. The trigger event when a user selects the Go button is: CSC_IB_TAB_ACTION_EVENT. Note that descriptions entered in the menu appear in the Actions LOV for agents to see.

2. Set the system profile, CSI: IB Tab - Action Menu, to the seeded or new menu. The profile's LOV displays user menu names of the registered menus.

   If you create multiple menus with different actions for different users, you can use this system profile to specify which responsibility sees which menu.

3. Use Form Personalization or the CUSTOM.pll to write the code. The trigger event when a user selects the Go button is: CSC_IB_TAB_ACTION_EVENT. Note that descriptions entered in the menu appear in the Actions LOV for agents to see.

To set up actions for the Orders tab:

1. Using Application Developer responsibility, create TSO functions (actions) for the Install Base tab and either add the functions to the seeded menu, ONT_ORDER_AGENT_ACTIONS (which has the user menu name of Sales Orders: Agent Actions) or create a new menu and add the new functions to it. Use Form Personalization or the CUSTOM.pll to write the code. The trigger event when a user selects the Go button is: OM_PRE_ACTION_EVENT. Note that descriptions entered in the menu appear in the Actions LOV for agents to see.

2. Set the system profile, OM: Contact Center Actions Menu Name, to the seeded or new menu. The profile's LOV displays user menu names of the registered menus. .

   If you create multiple menus with different actions for different users, you can use this system profile to specify which responsibility sees which menu.

3. Use Form Personalization or the CUSTOM.pll to write the code. The trigger event when a user selects the Go button is: OM_PRE_ACTION_EVENT. Note that descriptions entered in the menu appear in the Actions LOV for agents to see.

13

Set Up Oracle iStore

Overview of Set Up Oracle iStore Chapter

This chapter describes setup tasks that you must perform in Oracle iStore that are specific to the Oracle Telecommunications Service Ordering (TSO) solution.

Before You Begin

Before implementing Oracle iStore, you must set up the core applications for the Oracle TSO solution. For details, refer to the chapter, Implementing the Oracle TSO Solution, page 4-1.

Note: For Oracle Quoting integration, customer service representatives (sales agents) should only publish TSO quotes that are small enough to be handled via Oracle iStore. Since all TSO configurations are expanded in the cart, large configurations will not perform well, creating usability issues.

Setup Checklist

Following is the setup checklist for implementing TSO-specific functionality for Oracle iStore.

Oracle iStore-TSO Setup Checklist

Setup Step	Required/Optional
Implement Oracle iStore	Required
Set Up Payment Due with Order and Recurring Charges	Required

Setup Step	Required/Optional
Set Profile Options	Required
Set Up Items	Required
Perform Pricing and Payment Setups	Required
Perform User Setups	Required

Set Up Payment Due with Order

Payment Due with Order functionality allows implementers to display one-time (total net price) and recurring charges in the shopping cart and order review pages. For an example of how Payment Due with Order and recurring charges display in the Oracle iStore Customer Application shopping cart, Review Changes page, and Order Tracker, see the "Display of Totals Area", section in the chapter, Using the Oracle TSO Solution, page 16-1.

Every line in the shopping cart will be associated with one charge, either recurring (pay later amount) or one-time (pay now or pay later amount). For a given line, the system checks the periodicity of the item. If the periodicity is null, it means that the line does not have any recurring charges. If the periodicity is not null, then the recurring charge of the line is displayed in the appropriate column along with its periodicity.

In the Amount Due with Order area, the Sub-Total amount is calculated by an Oracle Receivables API and cannot be changed by the user. The Total Due With Order includes taxes and shipping/handling (unless setup is done to not show these). Only one payment method can be captured at the header level, and this payment method is used for the Payment Due with Order amount. The recommended payment instrument for pay now is credit card (since it allows authorization); however, this is not enforced in the code. Recurring charges lines should have pay later payment terms; otherwise, the pay now portion of these charges will be included in the Total Due Now amount.

Recurring items display as prices having a periodicity (e.g., Monthly, Quarterly, etc.). These items must belong to a network container. Assuming they are set up, recurring prices only display if the profile option, IBE: Enable Recurring Charges, is Yes. For each period, a sum of all charges is provided. Totals are displayed dynamically based on the items in the cart (e.g,. if there are no quarterly items, a quarterly total is not displayed).

Note: When setting up recurring charges, the value entered for each period is the value that displays in the shopping cart. Therefore, ensure that the word is entered in adjective form. For example, the period type Month should be entered as *Monthly* in order to display thusly in the shopping cart.

See "Set Profile Options", below, for more information on setting profile options to

enable Payment Due with Order and recurring charges.

Set Profile Options

Set the profile options in this section according to your business requirements.

IBE: Display Option Classes in the Shopping Cart and Order Tracker

This profile option specifies whether option classes are displayed in the Oracle iStore shopping cart, Order Tracker, and e-mail notifications. Set to Yes to enable the display. If set to No, option class children are displayed as direct children of the option class parent. Possible values: Yes/No. Default value: Null. If Yes or Null, option classes are displayed. This profile can be set at application, site, and responsibility levels.

IBE: Enable Install Base View

This profile option controls the appearance of the My Products subtab in Order Tracker in the Oracle iStore Customer Application. Set to Yes to enable the subtab. Possible values: Yes/No. Default value: No. See the *Oracle iStore Implementation and Administration Guide* (Implementing Carts and Orders chapter) for more information on the My Products page.

IBE: Enable Pay Now and IBE: Enable Recurring Charges

The profile option, IBE: Enable Pay Now, specifies whether Pay Now/Pay Later totals are displayed in the in the Oracle iStore Customer Application shopping cart and in the order review page. Set to Yes to enable the display of Pay Now/Pay Later totals. Possible values: Yes/No. Default value: No. This profile option can be set at application, site and responsibility levels. This profile option should be set to Yes only if the OM system parameter, Installment Options, is set to Enable Pay Now.

The profile option, IBE: Enable Recurring Charges, specifies whether recurring charges are displayed in the Oracle iStore Customer Application shopping cart and in the order review page. Set to Yes to enable the display. Possible values: Yes/No. Default value: No. This profile option can be set at application, site and responsibility levels. This profile option should be to Yes only if the Order Management Recurring Charge system parameter is set to Yes.

If IBE: Enable Recurring Charges is Yes, the shopping cart amount is removed from the Welcome Bin, regardless of the items present in the cart.

> **Note:** Note that the amount to be authorized is not controlled by these iStore profiles; these profiles only control the Oracle iStore user interface. The authorized amount is be based on OM system parameter, Installment Options.

In the case where IBE: Enable Pay Now is Yes and IBE: Enable Recurring Charges is No, the following will be displayed:

- Subtotal
- Shipping and handling
- Tax
- Total
- Amount Due With Order:
  - Subtotal
  - Shipping and handling
  - Tax
  - Total Due with Order

Profile Setup in Implementation with TSO and Non-TSO Sites

In an implementation where both TSO and non-TSO sites are set up, specific guidelines exist about the settings of the profile options, IBE: Enable Pay Now and IBE: Enable Recurring Charges.

Case 1: One site has both TSO and non-TSO stores

- The profile options should be disabled at site and application levels.
- The profile options should be enabled at responsibility level for the TSO stores only.

Case 2: One site contains only TSO stores

- In this case, the profile options should be set up at site level.

Case 3: One site contains only non-TSO stores

- In this case, the profile options should be disabled at all levels.

IBE: Autoquery Install Base for B2C Users and IBE: View Only Web Published Items in Install Base

IBE: Autoquery Install Base for B2C Users specifies whether an autoquery is performed when a B2C customer accesses the My Products tab in the Oracle iStore Customer Application Order Tracker. Set to No to enable search criteria in the UI; set to Yes to disable search criteria. For B2B users, the search criteria is always present; this profile option has no effect on B2B functionality. Possible values: Yes/No. Default value: Yes. This profile can be set at site, application, and responsibility levels.

IBE: View Only Web Published Items in Install Base specifies whether only Web

Published items are visible in the install base view in the My Products subtab in in the Oracle iStore Customer Application Order Tracker. Possible values: Yes/No. Default value: No. This profile can be set at application, site, and responsibility levels.

See the *Oracle iStore Implementation and Administration Guide* (Implementing Carts and Orders chapter) for more information on the My Products page.

IBE: Default Payment Term

Oracle iStore first checks the payment terms defined at item level, then, if it doesn't find an item-level payment term, it uses the profile option, IBE: Default Payment Term, to set the terms of the item. This profile option must be set to a proper value in conjunction with item-level payment terms. As a best practice, set this profile option to a Payment Due with Order payment term and then enable a Pay Later payment term at item level for each recurring charge. Alternatively, set this profile option to a Pay Later payment term, and then enable Payment Due with Order payment terms at item level for each Payment Due with Order item.

CZ: Publication Usage

When a customer configures a Container Model from the catalog in the Oracle iStore Customer Application, the usage set for the Model by the iStore Administrator in the iStore Administration UI is passed to Oracle Configurator. However, there may be times when a customer views the item in the shopping cart and clicks it to view the Item Details page. In this case, if the customer configures the item from the Item Details page, there is no longer a section context associated with the Model, and the usage will not be passed to Oracle Configurator. As a workaround for this potential issue, implementers can set the profile option, CZ: Publication Usage, at iStore application level to a valid usage (e.g., PHONE_FLOW or PLAN_FLOW) as stored in Configurator. Setting this profile will cause Oracle Configurator to pick up the usage of any Model (configured in the specific case noted here) from the profile option.

MO: Operating Unit and IBE: Item Validation Organization

In a TSO implementation, the profile options, MO: Operating Unit and IBE: Item Validation Organization, must be in sync at the responsibility level of the responsibility associated to the TSO specialty sites.

Set Up Items

Once the dependencies and profiles are set up, perform the setups in this section to set up the items for purchase through the Oracle iStore Customer Application:

Ensure Proper Setup of Items in Oracle Inventory

Ensure that your items are set up properly in Oracle Inventory, according to the

following guidelines:

- Assign periodicity to recurring items so that they can be priced accordingly. Existing items have a null periodicity by default. The periodicity of the item can be set either in Oracle Advanced Product Catalog or in Oracle Inventory.

  Note: Each item should only have one periodicity. The periodicity of an item should not be changed if a transaction exists for this item.

- For each item in a network container, set up item-level payment terms (item specific) in Oracle Inventory. This is done by selecting the payment term for each item in the Payment Terms LOV within the Master Items Invoicing tab.

- To be included in the amount to authorize in Payment Due with Order functionality, items must have their Invoiceable Item and Invoice Enabled flags enabled.

- Children of a configuration should not have their Inventory Web Orderable flags enabled, since they cannot be ordered standalone. The Web Orderable flag enables the Add to Cart button in the item details page of the Oracle iStore Customer Application. Such items also should be marked as Web Published so that they are retrieved by the search results in the Customer Application.

- Items that are not part of a network container, but which are associated with a periodicity, are not supported. Only children of a container should be associated with a periodicity.

- Payment Due with Order functionality is only supported for one-time lines. Recurring charges lines should have Pay Later payment terms; otherwise, the pay now portion of these charges will be included in the Total Due Now amount.

- A network container should not have any periodicity under the Order Management tab (and Payment Terms under Invoicing Tab) assigned to it.

For more information, see the chapters, Set Up Oracle Order Management, page 5-1, and Set Up Oracle Inventory and the Item Master, page 6-1.

Create Section Hierarchy

As a best practice, set up a node in the section hierarchy just for the TSO items. In Oracle iStore, create navigational sections as follows:

1. Parent section: Wireless Services; note that this can be a child of another section. For this section, assign the Display Template, Subsection List with Product Detail.

2. Child section of Wireless Services: Start with Phone

3. Child section of Wireless Services: Start with Plan

For both of the child sections (Start with Phone and Start with Plan) you may either leave the default Display Template (Product Detail) set or assign a different one, depending upon your business requirements. Note that all of these sections should be navigational sections.

Add TSO Items to Sections and Assign Usage

Using the Sections or Products pages, add the network container item to the two child sections (Start with Phone and Start with Plan) with the appropriate usages. For example, for the Start with Phone section, add the network container to it and assign the product the Start with Phone usage.

See the *Oracle iStore Implementation and Administration Guide* (Implementing Products chapter) for more information on item setup.

Perform Pricing and Payment Setups

In addition to setting up recurring prices in Oracle Advanced Pricing and Payment Due with Order functionality, perform the following pricing and payment setups:

- In the Oracle iStore Site Administration Application, assign price lists to the sites containing the TSO items.
- In the Oracle iStore Site Administration Application, assign credit card as a payment type to the sites containing the TSO items.

  Note: In a reconfiguration flow, if Credit Card is not available as a payment type on the site, the credit card will not get defaulted into the order, even if the original order was placed using Credit Card as the payment type. In this case, another payment type will be defaulted onto the order.

Perform User Setups

Perform the following user setups for Oracle iStore.

Ensure B2B User Permission

For B2B users only, TSO functionality is supported for users with the IBE_VIEW_NET_PRICE permission in their user role. B2C users do not have the concept of user permissions. If an Oracle iStore B2B user does not have the IBE_VIEW_NET_PRICE permission, only list prices are displayed -- and since in a TSO scenario there is no concept of a list pay now total -- users without this permission will not be able to see the proper prices.

Set Up Online Access to Existing Account

Customers who have already placed on order through a different channel than Oracle iStore and thus have a pre-existing relationship with your company can register to use Oracle iStore. This type of registration is known as Online Access to Existing Account. In addition to providing customers a quick and convenient way to access your online stores, by integrating directly into Oracle TCA customer database, it also helps to avoid the creation of unnecessary, duplicate accounts. The registration is accomplished by using existing Oracle Trading Community Architecture and/or Oracle Internet Directory (OID) user data and allowing the user to fill in the remaining necessary data. If no order has been placed by or for the user, the user will have to register as if he is a new customer, even if some data already exists for this customer.

Online Access to Existing Account functionality is supported for both B2C and B2B users. See the *Oracle iStore Implementation and Administration Guide* (Implementing User Management chapter) for more information.

14

Set Up Oracle Quoting

This chapter covers the following topics:

- Overview of Set Up Oracle Quoting Chapter
- Set Profile Options
- Set Up Recurring Charges
- Set Up Payment Due with Order
- Customize Quoting UI for TSO
- Design Tips

Overview of Set Up Oracle Quoting Chapter

This chapter describes setup tasks that you must perform to Oracle Quoting that are specific to the Oracle Telecommunications Service Ordering (TSO) solution. For more information on functionality, setting up, or using Oracle Quoting, refer to:

- *Oracle Quoting Implementation Guide*
- *Oracle Quoting User Guide*

Before You Begin

Before you can set up Oracle Quoting for use with the TSO solution, you must set up the core applications for the Oracle TSO solution. For details, refer to the chapter, Implementing the Oracle TSO Solution, page 4-1.

Setup Checklist

Following is the setup checklist for implementing TSO-specific functionality for Oracle Quoting.

Oracle Quoting-TSO Setup Checklist

Setup Step	Required/Optional
Implement Oracle Quoting	Required
Set Profile Options	Required
Set Up Recurring Charges	Required
Set Up Payment Due with Order	Optional
Customize Quoting UI for TSO	Optional

Set Profile Options

The following profile options pertain to TSO functionality with Oracle Quoting. For a complete list of mandatory Oracle Quoting profile options, see the *Oracle Quoting Implementation Guide*.

The following table summarizes the profile options and their recommended settings for TSO functionality.

ASO: Allow Quantity Updates for Component Item

This profile option determines whether users can change the quantity of a component. It applies to both the Forms and HTML versions of Oracle Quoting. If the setting is Yes, users can change component quantity. If the setting is No, users cannot change component quantity. You must set this option to No to prevent the user from changing the quantity of Component items of a Container Model.

ASO: Allow Quantity Updates for Top Level Model Item

This profile option determines whether users can change the quantity of a top-level model product. It applies to both the Forms and HTML versions of Oracle Quoting. You must set this option to No to prevent users from changing the quantity of the network container. If the setting is Yes, users can change the quantity of top-level model products. If the setting is No, users cannot update the quantity of top-level model products. If the setting is null, the default value is Yes.

ASO: Default IB Search Results

This profile option determines if instances appear by default in the search results form.

It applies only to the Forms version of Oracle Quoting. If the setting is None, no instances appear by default, and you must conduct a search in order to view any results. If the setting is null, the default value is None.

ASO: Default Install Base Relationship

This profile option determines the default Oracle Installed Base relationship for the Relationship Type menu on the Oracle Install Base Relationships page. It applies to both the Forms and HTML versions of Oracle Quoting. If the setting is Component_Of, the Relationship Type default value is Component-of. If the setting is Connected_To, the Relationship Type default value is Connected-To. If the setting is null, the default value is Component_Of.

ASO: Default Ordered Quantity in OC UI

This mandatory profile option determines the default quantity that appears in the QTY field of the Oracle Telecommunications Service Ordering window. It applies to both the Forms and HTML versions of Oracle Quoting. If the setting is null, a default quantity of 1 applies. You must set this value to 1 if you are using a container model.

ASO: Default Order Type

This mandatory profile option determines how to process the order in Oracle Order Management and Oracle Service Fulfillment Manager. You set up order types in Oracle Order Management, and you must include workflow to pass order to Oracle Service Fulfillment Manager. It applies to both the Forms and HTML versions of Oracle Quoting. If the setting is null, there is no assumed default value.

ASO: Default Quote Status

This mandatory profile option determines the default status of a new quote. It applies to both the Forms and HTML versions of Oracle Quoting.

ASO: Display Installed Base Attributes

This profile option determines whether or not attributes appear in the Installed Base search results for reconfiguration and trade-ins. It applies only to the Forms version of Oracle Quoting. If the setting is Yes, the attributes appear in the search results. If the setting is No, the attributes do not appear in the search results. If the setting is null, the default value is No.

ASO: Max Number of IB Search Results

This profile option determines the number of instances that can appear at one time in the component or connection details. It applies only to the Forms version of Oracle Quoting. If the setting is null, the default value is 100.

ASO: Use Network Container

This mandatory profile option determines if network container functionality is enabled. It applies to both the Forms and HTML versions of Oracle Quoting. If the setting is Yes, you can reconfigure TSO items. If the setting is No, you cannot. If the setting is null, the default value is No.

Set Up Recurring Charges

Oracle Quoting supports recurring charges for a product if the product has a charge periodicity and it is part of a Container Model. Quoting users should not add a product having a periodicity that is not part of a Container Model. If such a product is added to the quote, the user will not be able to submit the quote to an order in Oracle Order Management. This is due to the fact that Order Management will raise an error for such a scenario. The Order Management system parameter, Enable Recurring Charges, must be set to Yes in order to be able to place the quote (having recurring charges) as an order.

In Quoting HTML UI, the charge periodicities for recurring charges are displayed in the Charge Periodicity column. For a one-time charge, the value in the Charge Periodicity column is blank.

To enable the recurring charges feature in Oracle Quoting (HTML), implementers should expose the Charge Periodicity column from the Products subtab. For more information, see the section, "Customize Quoting UI for TSO", below.

Charge Periodicity and Quotes Total

The quote total will be the sum of the header level pricing charges and the quote lines (price, tax and charges) that do not have a periodicity. Thus, quote lines that have a periodicity (i.e., recurring charges) will not be included in the quote total. The following quoting functionality is based on this definition of the quote total:

- Searches against the quote total
- Order Amount qualifier (mapped to Quote Total)
- Seeded approval attribute, Quote Total
- For Contracts integration: Total article variable (mapped to Quote Total)

In addition, the following Oracle Order Information Portal attributes for Quoting (header level) exclude quote lines that have a periodicity:

- Total
- Total List Price

- Total Selling Price
- Total Adjustment Amount
- Total Adjustment Percent

Derivation of Charge Periodicity

Following is the impact of the profile option, ASO: Enable Defaulting Rule, on how charge periodicity is derived:

At the time of quote line creation:

- If the profile option is Yes, then the default value for the Charge Periodicity is derived per the defaulting rule.
- If the profile option is No, then the default value for the Charge Periodicity is derived from the item setup in the Oracle Inventory Item Master.

At the time of quote template line creation:

- The default value for the Charge Periodicity is derived from the item setup in the Oracle Inventory Item Master.

Set Up Payment Due with Order

For payment terms that have one or more payment installments, the Days Due field must be set to 0 (zero). In addition:

- If such a payment term is associated to a quote line, then the quote line will have a non-zero pay now amount (assuming a non-zero quote line price).
- If such a payment term is associated to the quote header and there are header level pricing charges on the quote, then there will be a non-zero pay now amount associated with the quote header level pricing charges.

Implementers may wish to alter the Oracle Quoting UI (for both Forms and HTML) to display elements specific to the pay now amount. For more information, see the section, "Customize Quoting UI for TSO", within this chapter.

Customize Quoting UI for TSO

Both the HTML and Forms version of Oracle Quoting allow implementers to configure the UI to support TSO-related fields and UI elements. These setups are discussed below.

Modifying the Products Page in HTML Quoting

In the HTML version of Oracle Quoting, implementers can change the display of

columns on the Products page to expose TSO-related columns. The following list shows a sample setup of the columns implementers might display on the Products page:

- Charge Periodicity
- Instance Name

Also note the following:

- The Template Details page can display the Charge Periodicity column.
- The display of the Instance Name is not dependent on the profile option, ASO: Use Container Model setting. Implementers can show/hide Instance Name using OA Personalization. This applies wherever Instance Name is displayed in the HTML BLAF UI.
- When using the TSO solution, implementers may wish to rename columns to be more applicable to the TSO functionality.

For more information, refer to the *Oracle Quoting Implementation Guide* chapter, Implementation Tasks for Oracle Quoting (see the section, "Enabling Oracle Quoting Features with OA Personalization").

Modifying the Pricng Tab in Forms Quoting

With the Forms version of Oracle Quoting, implementers can customize the Pricing region. When using the Oracle TSO solution, implementers may wish to expose the Charge Periodicity column.

For more information, refer to the *Oracle Quoting Implementation Guide* chapter, Implementation Tasks for Oracle Quoting (see the section, "Using Folder Functionality to Customize Forms").

Design Tips

Consider either of the following for the default Line Type:

- An Oracle Configurator Functional Companion must always set the default Line Type.
- The default Line Type should be blank in Oracle Quoting, and the Oracle Order Management defaulting rules in the order (as opposed to the quote) set the default Line Type.

Use the selectable columns OA personalization feature to show the Line Type on the Quoting lines page and to rename columns, such as renaming Line Type to Action.

Use lookups to disable line and table-level actions that do not apply to the given implementation, such as Split Line, Add Service, ATP check, or Trade In option in

Installed Base searches.

For more information, see the *Oracle Quoting Implementation Guide*.

15

Set Up Oracle Service Fulfillment Manager

This chapter covers the following topics:

- Overview of Set Up Oracle Service Fulfillment Manager Chapter
- Create SFM-Enabled Workflow Header Process
- Create SFM-Enabled Workflow Line Process
- Add Function for Menu Notifications
- Create PL/SQL Procedure to Update Active Flag
- Create Workflow Process
- Define Work Item
- Map Line Types
- Enable Order Cancellation
- Design Tips

Overview of Set Up Oracle Service Fulfillment Manager Chapter

This chapter describes setup tasks that you must perform to Oracle Service Fulfillment Manager (SFM) that are specific to the Oracle Telecommunications Service Ordering (TSO) solution.

Before You Begin

Before you can set up Oracle Service Fulfillment Manager for use with the TSO solution, you must set up Oracle Installed Base and Oracle Order Management. For more information, see the chapter, Set Up Install Base.

Setup Checklist

Following is the setup checklist for implementing TSO-specific functionality for Oracle

Service Fulfillment Manager.

Oracle Service Fulfillment Manager-TSO Setup Checklist

Setup Step	Required/Optional
Implement Oracle Service Fulfillment Manager	Required
Create SFM-Enabled Workflow Header Process	Required
Create SFM-Enabled Workflow Line Process	Required
Add Function for Menu Notifications	Optional
Create PL/SQL Procedure to Update Active Flag	Optional
Create Workflow Process	Optional
Define Work Item	Optional
Map Line Types	Optional
Enable Order Cancellation	Optional

For more information on functionality, setting up, or using Oracle Service Fulfillment Manager, refer to the *Oracle Service Fulfillment Manager Implementation Guide*.

Create SFM-Enabled Workflow Header Process

Use the following procedure to create an Oracle Order Management Workflow header process to use for SFM. In this procedure, log in to Workflow Builder.

Steps

1. Open the OM Order Header workflow item type (OEOH).
2. Open the Service Fulfillment Header workflow item type (XDPOMOH).
3. In the item type, OEOH, open the Order Flow - Generic process.
4. Save this process with a new name Order Flow - Service Fulfillment.

5. Drag and drop the Service Fulfillment Order process from the item type, XDPOMOH, into Order Flow - Service Fulfillment between the Book - Order, Manual and Close - Order processes.

6. Save your work.

Create SFM-Enabled Workflow Line Process

Use the following procedure to create an Oracle Order Management Workflow line process to use for SFM. In this procedure, log in to Workflow Builder.

Steps

1. To customize the Workflow, change the Access Level to 100:
   - Choose Help, About Oracle Workflow Builder.
   - In the Access Level field, enter 100.
   - Click OK.
2. Open the OM Order Line workflow item type (OEOL).
3. Open the Service Fulfillment Line workflow item type (XDPOMOL).
4. In item type OEOL, open the Line Flow - Service Fulfillment process that you created in the section, "Add Installed Base Activity to OM Line Workflow", in the chapter, Set Up Order Management.
5. Drag the activity, Service Fulfillment (For Activation Required Items), from item type, XDPOMOL, and drop it between Ship - Line, Manual and Fulfill - Deferred.
6. Save your work.

Add Function for Menu Notifications

This is an optional procedure.

In this procedure you are adding the Workitem Notification Redirection to the menu for notifications. This procedure assumes that you have not already added Workitem Notification Redirection to the menu. This function lets you view error notifications in HTML. For more information, see the section, "Manage Failure Notifications in HTML", in the chapter, Administering the Solution.

In this procedure, log in to Oracle Forms with System Administrator responsibility and navigate to Applications, Menu.

Steps

1. Select Menu and click Open. The Menus window opens.
2. In the Menu field, query for FND_WFADMIN_NEW.
3. Select New.
4. On the new empty line, keep the Prompt and Submenu fields empty.
5. In the Function field, click the List of Values. The Function window opens.
6. Enter the Search criteria, %redirect%.
7. Select Workitem Notification Redirection, and click OK. Both the Function and Description fields automatically populate.
8. Save your work.
9. When the message about recompiling menus appears, click OK.

See also: The end-user procedure that uses the preceding setup is in the section, "Manage Failure Notifications in HTML", in the chapter, Administering the Solution

Create PL/SQL Procedure to Update Active Flag

The following information is a specific example that you can consider for a reference. It is not a requirement that you create your PL/SQL procedure in this manner.

In this procedure, use the PL/SQL editor of your choice to create a package in your instance. The procedure in this package updates the work item parameter, ACTIVE_FLAG, to Y. You define ACTIVE_FLAG as an extended attribute for your Service Fulfilment Items in the section, "Map Work Item to Item and Action (Line Transaction Type) Combination", below.

```
CREATE OR REPLACE PACKAGE XXMACD_SFM_UTILS AS
PROCEDURE SET_ACTIVE(itemtype IN VARCHAR2,
itemkey IN VARCHAR2,
actid IN NUMBER,
funcmode IN VARCHAR2,
resultout OUT VARCHAR2);
END;
/
CREATE OR REPLACE PACKAGE BODY XXMACD_SFM_UTILS AS
PROCEDURE SET_ACTIVE(itemtype IN VARCHAR2,
itemkey IN VARCHAR2,
actid IN NUMBER,
funcmode IN VARCHAR2,
```

```
resultout OUT VARCHAR2) IS
l_wi_instance_id NUMBER ;
BEGIN
-- RUN mode - normal process execution
--
IF (funcmode = 'RUN') THEN
l_wi_instance_id := WF_ENGINE.GetItemAttrNumber
(itemtype => itemtype ,
itemkey  => itemkey ,
aname => 'WORKITEM_INSTANCE_ID');
XDP_ENGINE.SET_WORKITEM_PARAM_VALUE
(P_WI_INSTANCE_ID => l_wi_instance_id,
P_PARAMETER_NAME => 'ACTIVE_FLAG',
P_PARAMETER_VALUE   => 'Y');
resultout:= 'COMPLETE';
RETURN;
ELSE
resultout := xdp_om_util.HandleOtherWFFuncmode(funcmode);
RETURN;
END IF;
EXCEPTION
WHEN OTHERS THEN
wf_core.context('XXMACD_SFM_UTILS','SET_ACTIVE', itemtype, itemkey,
actid, NULL);
RAISE;
END SET_ACTIVE;
END XXMACD_SFM_UTILS;
```

Create Workflow Process

The following example is an option. It is not a requirement.

This example assumes sample names for:

- **Item Type:** SFMTEST - SFM Test Item Type
- **Function:** Update WI Active Flag
- **Procedure:** XXMACD_SFM_UTILS.SET_ACTIVE
- **Process:** SFM MACD Test WorkItem Process

In this procedure, use Workflow Builder module.

Steps

1. Create a new Item Type. For example: SFMTEST - SFM Test Item Type.
2. Associate the following three previously defined Service Fulfillment Manager item types to the new item type:
   - SFM Lookup Codes
   - SFM Standard
   - Standard
3. Create a Function. For example: Update WI Active Flag.
4. Map the function to the procedure, XXMACD_SFM_UTILS.SET_ACTIVE, that you created in the section, "Create PL/SQL Procedure to Update Active Flag", above.
5. Map Error Item Type to XDPWFSTD.
6. Map Error Process to XDP_ERROR_PROCESS.
7. Define a process. For example: SFM MACD Test WorkItem Process.
8. Drag the function, Update WI Active Flag, into the process.
9. Drag the functions, Start and End, from Item Type Standard and into the process.
10. Drag the function, Complete Work Item, from Item Type SFM Standard into the process.
11. Define the flow between the functions as follows:
    - Start...to
    - Update WI Update Flag...to
    - Complete Work Item...to
    - End
12. Save your work.

Define Work Item

The following example is an option. It is not a requirement.

This example assumes sample names for:

- **Item Type:** SFMTEST - SFM Test Item Type
- **Function:** Update WI Active Flag
- **Procedure:** XXMACD_SFM_UTILS.SET_ACTIVE
- **Process:** SFM MACD Test WorkItem Process

In this procedure, log in to Oracle Forms with SFM System Administrator responsibility and navigate to Setup, Service Definition, Work Items.

Prerequisites (see the following sections:)

- "Create PL/SQL Procedure to Update Active Flag"
- "Create Workflow Process"

Steps

1. For the work item, specify Display Name (Set Active), Internal Name, Version and Begin Date.
2. From the list, choose Work Item Type User-Defined Workflow.
3. From the Item Type list, choose the SFM Test Item Type that you created in the section, "Create Workflow Process", above.
4. From the Process list, choose the SFM MACD Test WorkItem Process that you created in the section, "Create Workflow Process", above.
5. Specify an Item Key for this Work Item. For example, an Item Key could be SFMMACD. The Item Key value becomes the prefix for the Workflow Item Key.
6. Navigate to the Parameters tab.
7. Choose the parameters: Active, Phone, and Primary. These parameters are extended attributes that you defined in the section, "Create Extended Attributes", in the chapter, Set Up Install Base.

Map Line Types

You can attach one or more work items or tasks to a combination of item and action. In this procedure, log in to Service Fulfillment Manager with SFM Administrator responsibility and navigate to Setup, Service Definition, Service Actions.

Prerequisites

You must have created the work item, the item, and the action.

Steps

1. For each of the Service Fulfillment items that you defined according to standard setup for Oracle Service Fulfillment Manager, navigate to the Actions tab and map the Line Transaction Types, ADD and CHANGE, to the item. For more information on setting up Service Fulfillment items, see *Oracle Service Fulfillment Manager Implementation Guide.*

2. Navigate to the Work Item for Action tab and map Work Item Set Activate to the two Item Action combinations.

Enable Order Cancellation

To set up order cancellation, you must:

- Create a validation template in Oracle Order Management
- Create a Processing Constraint
- Build Rollback Activities into Workflow
- Test Order Cancellation

These steps are discussed below.

Create Validation Template

The following procedures describe how to create a validation template in Oracle Order Management for the purpose of setting up order cancellation in Oracle Service Fulfillment Manager.

In this procedure, log in to Oracle Order Management and navigate to Setup > Rules, Security, Validation Templates.

Steps

1. In the Entity field, specify Order Header.
2. Enter the Template Name. For example: Validate SFM Cancel.
3. Enter the Short Name. For example: SFMCNCL.
4. Enter Description. For example: Attempt to cancel order in SFM.
5. Select Validation Type as the API.
6. Enter the PL/SQL Package Name, XDP_CANCEL_ORDER.
7. Enter the Procedure Name, CANCEL_SFM_ORDER.

8. Save the validation template and record the name of the template.

Create a Processing Constraint

After you have created a validation template, create a processing constraint to enable order cancellation in Oracle Service Fulfillment Manager.

In this procedure, log in to Oracle Order Management and navigate to Setup > Rules, Security, Processing Constraints.

Prerequisite

Validate template has been created.

Steps

1. Query for the Application, Order Management, and the Entity, Order Header.
2. For the Constraint Cancel, add the following condition:
    1. Navigate to Condition tab.
    2. Select the Scope, Any.
    3. In Validation Entity field, enter Order Header.
    4. In the Record Set field, enter Order.
    5. Enter the validation template name that you specified when you created the validation template above.
    6. Enter a User Msg that should appear when the order cannot be cancelled.
3. Save the Processing Constraint.

Note: The condition for the Cancel Order functionality should always be the last condition.

Build Rollback Activities into the Workflow

To enable order cancellation in Oracle Service Fulfillment Manager, you must also build Rollback Activities into the workflow. This is an implementation dependent task.

An SFM_Rollback Workflow has seeded:

- Item_Type = XDPPROV (SFM Fulfillment Processes)
- Dummy Process = CANCEL_ORDER_PROCESS

You can extend this process at implementation time to do business-specific rollback for any Oracle Service Fulfillment Manager work done up to the point that the cancel order

was executed.

Test Order Cancellation

Test that order cancellation is working properly. You can cancel orders only at header level and not at the line level.

In this procedure, log in to Oracle Order Management and navigate to Order, Returns, Sales Order.

Steps

1. Query the order you wish to cancel.
2. At the order header level, select Action.
3. From the List of Actions, select Cancel.
4. Enter a Reason for cancellation and any comments.
5. Click the OK button.

Note: If cancellation is successful, the Rollback Workflow starts, or otherwise, the user-defined error message appears.

Design Tips

Some tips relating to Oracle Service Fulfillment Manager include:

- An attribute that is mandatory in Oracle Service Fulfillment Manager also needs to be mandatory in Oracle Configurator Developer. For more information about configuration attributes, see *Oracle Configurator Methodologies*.
- If Oracle Service Fulfillment Manager fails to accept the order after booking the order in Oracle Quoting, then Oracle Quoting never receives the error message. In Oracle Order Management's Sales Order form, these errors have a status of Provisioning Failed. To see the status of these provisionable order lines, you must go to the Oracle Order Management Sales Order form.
- If Oracle Service Fulfillment Manager fails to accept the order after booking the order in Oracle Quoting, you must progress the order from Oracle Order Management's Sales Order form.
- The Oracle Bills of Material routing includes the definition of the activation sequence of lines. When Oracle Service Fulfillment Manager receives a provisionable model with provisionable components, Oracle Service Fulfillment Manager provisions the model first and then its components.

- The model developer must ensure that:
  - Every provisionable item includes a mapped Line Type in Oracle Service Fulfillment Manager.
  - The item and Line Type combination maps to a Workitem.
  - When order booking occurs, Oracle Service Fulfillment Manager receives all the mandatory Workitem parameters (also known as extended attributes) with a value.

16

Using the Oracle TSO Solution

This chapter covers the following topics:

- Overview of Using the Oracle TSO Solution Chapter
- Using Oracle Contact Center
- Using Oracle iStore
- Using Oracle HTML Quoting
- Using Oracle Forms Quoting
- Using Oracle Order Management

Overview of Using the Oracle TSO Solution Chapter

This chapter contains user tasks specific to the Oracle Telecommunications Service Ordering (TSO) solution.

Using Oracle Contact Center

This section contains information that can aid sales agents in using TSO-specific functionality with Oracle Contact Center. Oracle Order Management can be launched from within Oracle Contact Center and Oracle TeleService. Refer to the product documentation for these applications for information generic to these applications and to Oracle Contact Center. See the Preface, page xii of this guide for a list of relevant documentation.

Overview of Contact Center Windows

In the Oracle Contact Center Orders and Install Base tabs, sales agents can configure and reconfigure Container Models.

Orders Tab

The Orders tab in the Contact Center window enables sales agents to search for and select TSO items, to configure those items, and to place orders with the items. Note the following about the Orders tab:

- The Price Summary subtab (enabled only if the Line Items sub-tab contains entered line items) displays the charges (one-time or recurring) details and the first installment details for TSO items.
- The Orders tab includes only one seeded action: Add Item Instance. This permits agents to add new instances on the Create Item Instance page of Oracle Installed Base. Additional actions defined by the implementer also will display.
- Defaulting rules set up in Oracle Order Management apply in the Orders tab.
- Sales orders created in Oracle Contact Center follow the TSO flow, where actions are mapped to line types, and the line types drive the workflows for the lines in fulfillment phase. The workflow started for a line is derived based on the line type and item type code combination.
- Sales agents can launch Oracle Configurator from the Orders tab by selecting the Configurator button on TSO-enabled items.
- In the Orders tab, sales agents must display the order details by drilling down on an order in the tab and then click the Actions button (highlighted in the image above).
- Implementers who wish to extend or disable any of the Oracle Order Management actions, they must do so separately according to the procedures described in that product's documentation.

Install Base Tab

The Install Base tab in the Contact Center window enables sales agents to search a customer's install base and reconfigure existing TSO items. Note the following about the Install Base tab:

- The Add to Existing Order checkbox, if checked, adds the change to the item instance as a new order line. If unchecked, a new order is created for the item instance.
- The Add to Existing Order checkbox is enabled when the action, Add from Install Base, is selected from the Orders - Line Items subtab page. In all other cases, it remains unchecked.
- If a sales agent adds the instance to the existing order, and the customer is making a

change to an existing item instance, the item instance is added to any Container Model already existing as an order line on the order. If the item instance cannot be added to an existing Container Model, a new Container Model is created.

Install Base Tab Actions

The application ships with the following actions available to agents in the Install Base tab:

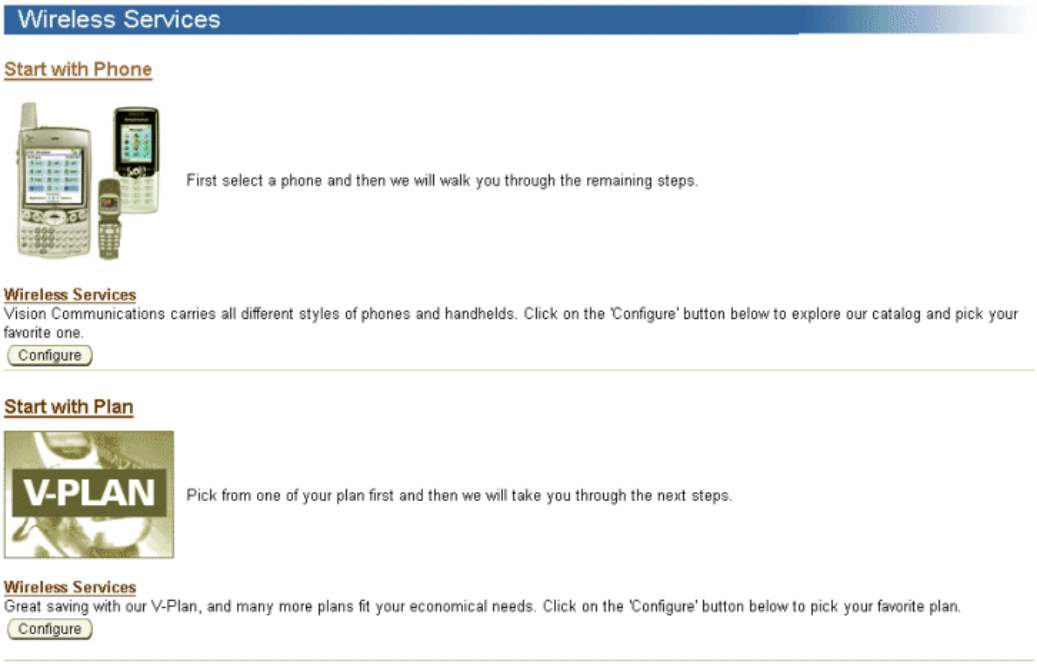
- **Disconnect:** Intended for telecommunications customers only: Creates an order to disconnect telephone service.
- **Update Item Instance:** Displays in a new browser window the item in the Item Instance Details page (Oracle Install Base) where the agent can make updates.
- **Reconfigure a MACD Item Instance:** Intended for telecommunications customers only; brings up Oracle Configurator, where agents can reconfigure the customer telephone services.
- **Create Service Request:** Opens up the Service Request tab of the Contact Center and populates a new service request with the selected installed base item.
- **New Item Instance:** Displays in a new browser window the Create Item Instance page (Oracle Install Base).

Using Oracle iStore

Catalog Display of TSO Items

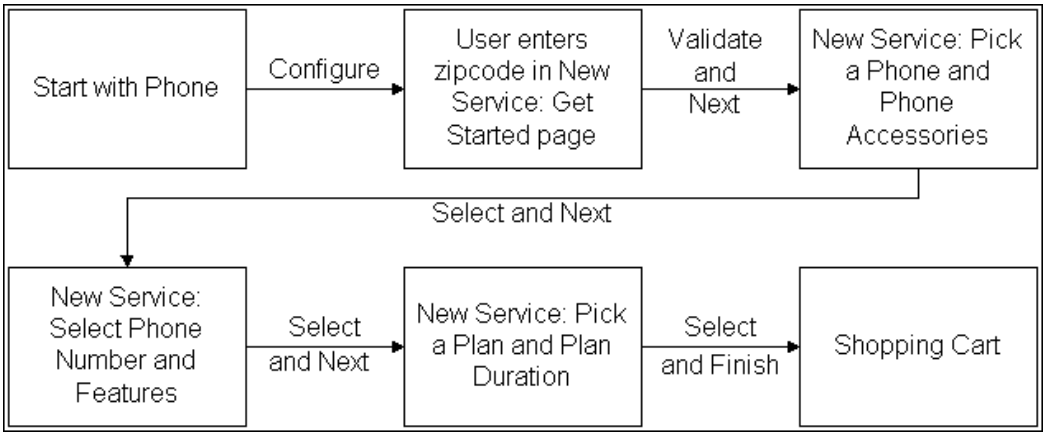
In a typical implementation, the merchant sets up a navigational parent section (named, for example, Wireless Services) which contains two navigational subsections (e.g., Start with Phone and Start with Plan) containing the wireless components. This allows the customer to start the configuration either from the plan or the phone and then proceed to the next logical step. Oracle iStore provides Display Templates for the recommended flow (see flows below). If the customer starts with the phone, he will be taken to the plan step next, and vice versa. The graphic below shows how the sections might look in an implementation of Oracle iStore.

Sections with TSO Items Example



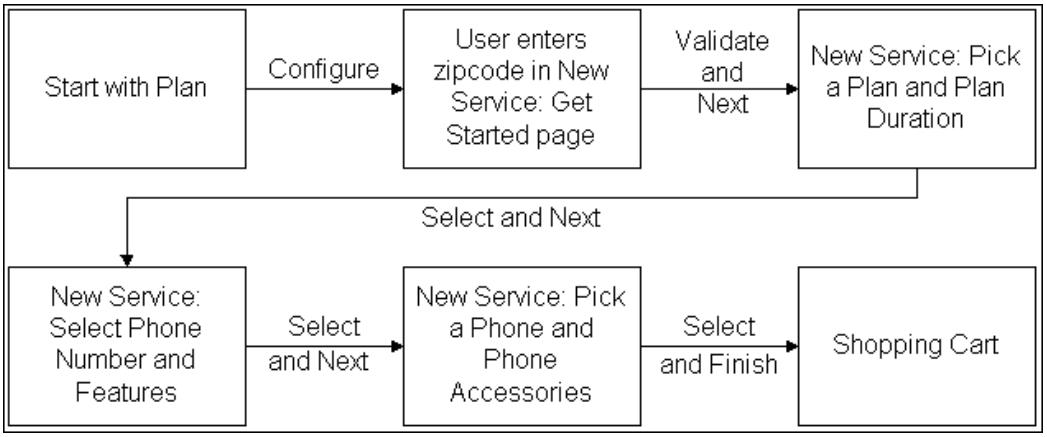
The following graphic shows the flow for the Start with Phone setup.

Catalog Flow for Start With Phone



The following graphic shows the recommended flow for the Start with Plan setup.

Catalog Flow for Start with Plan



While going through both of the above flows, a breadcrumb trail is presented in the user interface. This trail is provided by Oracle Configurator.

Rules and Guidelines for Catalog Display

Following are rules and guidelines for the display of TSO items in the catalog (note that many of these rules are determined by the setup in Oracle Configurator):

- Oracle Configurator does not ship any Models or usages for the Oracle iStore TSO catalog flow. These usages must be created in Oracle Configurator Developer.
- During the configuration, if the customer clicks on a Customer Application catalog tab or a top-level icon, the configuration ends and is not saved. However, a line with the network container will be present in the cart with an incomplete icon.
- Based on Configurator rules, components of a configuration might be automatically selected based on choices done in a previous step of the configuration process. For example, if the customer selects the 300 Minutes National Plan, caller ID might be an included feature and would then be pre-selected.
- Selections displayed in a configuration step are restricted based on the selections done in the previous step (based on configuration rules). For example, if the customer selects an X phone in the first configuration step, then in the second configuration step, only X plans (plans that are compatible with the selected phone) are displayed.
- By default, support is provided for validations (configuration rules) existing within the same page.
    - Example 1: Based on the plan selection, three features out of the 10 available features are displayed on the same page and can be selected.

    - Example 2: Some features are automatically selected based on other selections on the same page. For example, if call blocking is selected, caller ID is automatically selected.

- If the implementer adds a component (whether a leaf component, middle-level, or top-level) to the Bill of Material (BOM), the new component is displayed appropriately in the Customer Application.

- The ability to capture attributes is supported in the Configurator UI. An example of an attribute is the phone number for the call forwarding feature.

- Support for family plans: BOM models support the ability to have one plan connected to multiple phones and multiple sets of features. In the Customer Application, the user starts first by choosing the plan and then can add up to X phones (and the same number of feature sets) total on the family plan.

- List prices are displayed for each component on configuration pages. For a MACD implementation with recurring charges, no prices are displayed in the summary page, since the totals would be incorrect. Since selling prices can only be displayed in the summary, selling prices are not displayed in the Configurator UI. In addition, periodicity is not displayed in the Configurator portion of the user interface.

- Children of a configuration are searchable from the category search and the section search in the Customer Application. However, after retrieving a child item in the search and then drilling down to the item details, the user cannot add the item to the cart (at least in a recommended setup where these items do not have their Inventory Web Orderable flag enabled, since they cannot be ordered standalone). Such items should be marked as Web Published so that they are retrieved by the search results. From the search results, the user can navigate to the section where this network container (parent of the item searched) is published. This is done via a link in the Display Template.

- For network containers, the Configure Your Product bin in the item details page does not show Quantity, since the quantity of a network container is always 1. This is also true of network container display in the catalog.

- In the two Display Templates, the quantity is hidden and is always 1 for a network container. There is no change in behavior for standard configurations. The templates are: Product Multi-Select with Drill-Down and Subsection List with Product Detail.

- Related items can be displayed on the item details and shopping cart pages.

- For standard items, in the item detail page, the Add to Cart bin displays with Price, Quantity, and an Add to Cart button. For network containers, the price and quantity field is hidden, and only the Configure button displays.

Shopping Cart Display of TSO Items

Following is the behavior of TSO items in the shopping cart:

- Customers can initiate configuration changes from the shopping cart. Once the configuration is complete and being viewed in the shopping cart, the user can initiate a change at the network container level. This will take the user back to the Configurator UI within the Customer Application.

- After the customer is done selecting products, he can see all of the products and their prices in the shopping cart. The prices are summarized by period (one-time, monthly, quarterly, etc.). The customer cannot change the periodicity from the shopping cart. A given line of the shopping cart will have either a one-time price or a periodic price (only one periodicity per line), but cannot have both.

- Only the top-level model is displayed in the shopping cart, along with its rolled-up price. The Pricing Details page available from the cart page shows only total discounts (not including freight and special charges).

- Configuration details are displayed in an in-line page. The in-line page only has one price column, with a price for every line. For every line, the Pricing Details pop-up page shows discount details. On the top model line, the prices for the children in the configuration are not displayed. The total price at the bottom is the model plus children.

- Cart display for network containers: The configuration displays as expanded in the cart, and prices are never rolled up at the top model level. Instance names are displayed for all lines of the container (whenever available from Oracle Quoting). For models, sub-models and option classes, the price and taxes are not displayed if list price is zero.

- Cart display for BOM bundles: Model bundles are displayed like standard configurations. In the in-line page, no prices are displayed, and no Price column displays.

- Displaying option classes in the shopping cart is dependent on the profile option, IBE: Display Option Classes in the Shopping Cart and Order Tracker. If an option class is not displayed, its children are displayed as direct children of the option class' parent. This also applies to Order Tracker and e-mail notifications.

- Discounts display: Discounts for recurring lines are displayed similarly to one-time lines (i.e., values are displayed in brackets if negative, and the corresponding percentage is displayed in parentheses). For recurring lines, discounts and surcharges have the same periodicity as the lines. In the Pricing Details pop-up page, the periodicity is displayed for prices, discounts, and surcharges.

- If a quote is published with an item-level payment instrument, the user can modify

only the header (cart-level) payment instrument. The header payment instrument does not overwrite the item-level payment instrument.

- Children of a configuration that are currently published in Oracle iStore (i.e., the Web Published flag is enabled) are hyperlinked in the shopping cart. The hyperlink will take the user to the item detail page.

- From the shopping cart, the customer can change his services and shippable items. The Reconfigure button takes the user to first step of the configuration (based on the BOM sequence). The Reconfigure button is available only at the network container level.

- In the shopping cart and on the order review page, and when uploading a cart via Direct Item Entry:
  - The quantity of network container should be 1.
  - The quantity of the root model (MI, PTO model) will always 1 and will be non-shippable.
  - The quantity of tangible items cannot be more than 1.

- When the cart template is configured to display only subtotals (no taxes, shipping/handling, totals), the pay now area also displays only the subtotal.

- From the shopping cart (in an initial configuration flow), the customer can add an extended warranty to any item (shippable or non-shippable), which is serviceable and the child of a network container. The customer also can remove an extended warranty he has just added.

- Terms and conditions can display based on the content of the cart.

- Specific implementation steps are required to pick up the usage of a Model when a customer configures a TSO item from the Item Details page after navigating to this page from the shopping cart.

The following graphic shows how TSO items in the shopping cart might look in an implementation of Oracle iStore.

Example Shopping Cart Display of TSO Items

Your Oracle Store cart expires on 01-Aug-2002. Please save this cart if you intend to use it beyond this date.

Part Number	Item Name	UOM	Quantity	Price	Remove
CM34521	Wireless Service Reconfigure	Each	1		
CM34521	Phone: 650 345 6703	Each	1		
CM34521	palmOne Treo 600 Add Service	Each	1	$29.99	
	VAT (Rate:17.5%)			$10.00	
CM34521	1 Year Silver Warranty	Each	1	$10.00	
	VAT (Rate:17.5%)			$10.00	
CM34521	Plan	Each	1		
CM34521	Nationwide 250	Each	1	$35.00 Monthly	
	VAT (Rate:17.5%)			$10.00	
CM34521	Activation Charge	Each	1	$29.95	
	VAT (Rate:17.5%)			$10.00	
CM34521	Features: 650 345 6703	Each	1		
CM34521	Voice Mail	Each	1	$9.95 Monthly	
	VAT (Rate:17.5%)			$10.00	
CM34521	Call Forwarding	Each	1	$4.95 Monthly	
	VAT (Rate:17.5%)			$10.00	

Review Changes Page Behavior and Guidelines

The following behavior and guidelines apply to the Review Changes page. Note that the Review Changes page will only appear in a reconfiguration flow.

- The billing/shipping information is defaulted with the primary billing/shipping address. If there is no primary information, the user is shown a message in the page and can add a billing/shipping address.

- For every line the action (Add, Move, etc.) is displayed in the Action column. Disconnected lines are displayed with a null price, if the pricing setup indicates so. In most cases, implementers should enforce such a pricing setup (set up disconnected lines to have a null price) to avoid unnecessary charges applied to disconnected lines. Changed lines are marked with the new price.

- The untouched (non-reconfigured) lines are handled differently depending on the Oracle Configurator profile option, CZ: Include Unchanged Install Base Items. If the profile is set to False, all of the untouched lines are removed from the cart. If the profile is set to True, the following will happen: (1) Oracle iStore hides (shippable and non-shippable) non-recurring, untouched lines; (2) Recurring untouched lines are left in the reconfiguration cart; (example: Voice Mail - $5/month).

    - Here are some possible business needs for keeping recurring lines: (1) Prices might have changed and the customer needs to be informed; and (2) Price changes, if any, need to be notified to the billing system.

- The customer has the option of adding extended warranties from the Review

Changes page, if available.

- If a B2C customer selects Cancel in the Review Changes page, he is taken back to the My Products page. If a B2B user selects Cancel, he is returned to the configuration details page, where the Reconfigure button is displayed.

- For published quotes which are reconfiguration quotes, whether or not the customer can update them is dependent upon whether quote update is allowed (same as standard quotes). However, when a reconfigured quote is activated, it does not become the active cart, as in standard quotes. Instead, the customer is taken to the review T&Cs page (if T&C functionality is implemented) and then directly to the configuration UI). Standard items, in this case, cannot be removed or added.

- The sources for the data displayed in the columns are as follows:
  - **Action column:** This is the line-level action of the reconfiguration, sourced from the quote line type. Note that this column is populated only for children of a network container and the network container itself. This column only appears during a reconfiguration flow.
  - **Part Number column:** This is the part number of the item in Inventory.
  - **Item Name column:** This is the item short description from Inventory concatenated with the instance name. These two elements are separated with a colon (:) sign.
  - **UOM column:** This is the unit of measure for the item as defined in Inventory.
  - **Quantity column:** The quantity for a network container and all of its children is read only and always equal to 1.
  - **Price column:** The price column displays the unit net price of the item multiplied by the quantity. If the item is periodic, the periodicity will be displayed along with the price of the item. All prices (not including tax lines) are hyperlinked, allowing the user to retrieve pricing details.

- Note the following additional behavior of other elements of the page:
  - **Reconfigure link:** The Reconfigure icon is only present for top-level models (network container or not). If the customer selects this icon, he is taken to the first step of the BOM.
  - **Details/Reconfigure link:** This link displays only for top-level models that are not network containers. Selecting the link takes the user the configuration details page.

- **Add Service icon:** This icon allows the user to add a service to the item. The icon is present only on Add lines.

- **Remove button:** The remove icon does not display for children of a network container. In this scenario, the Remove button can only show up for an extended warranty.

Display of Totals Area

For TSO items, the totals of the pay now and recurring charges display at the bottom of the Shopping Cart and Review Changes pages.

- **One Time Price:** This line displays the sum of the one time charges for the entire cart.

- **Periodic Charges:** This line displays the sum of the periodic charges (e.g., Monthly, Quarterly). If there is more than one sum by period, then multiple period charges display.

- **Sub-Total:** This line displays subtotals of prices that need to be paid at the time of ordering.

- **Tax:** This line displays taxes that need to be paid at the time of ordering.

- **Shipping and Handling:** This line displays shipping and handling charges that need to be paid at the time of ordering.

- **Total Due with Order** This line displays the total amount that needs to be paid at the time of ordering. This amount will be authorized on the credit card if credit card is the payment method specified. Taxes are calculated for the subtotal of the charges due with order. Shipping and handling are calculated only for the pay now lines.

The following graphic shows how the one-time and recurring charges might display in an implementation of Oracle iStore.

Payment Due with Order and Recurring Items Example

Order Tracker Display of TSO Items

TSO display and usage in the Order Tracker area of the Customer Application is described in the Oracle iStore Implementation and Administration Guide (Implementing Carts and Orders chapter).

Note: Network container Items and their child items are not returnable.

Reconfiguration Behavior with TSO Items

Customers can reconfigure network container items (including services and/or shippable items) by selecting the Reconfigure button on applicable items from the Configuration Details page (My Products page, Details link). This action ultimately leads to the Review Changes page where the user can place the order (see the process flows section above for more information on the basic reconfiguration flow). The Reconfigure button is displayed only if all of the below conditions are met:

- The parent is a network container
- The network container is Web Published
- The network container is Web Orderable
- The network container is published in the site the user is currently in

Note that the network container is displayed even if only part of the configuration is Install Base-trackable (IB-trackable), assuming the above conditions are met.

For example, in the case below, Network Container A would still be displayed on top of Item 2:

- Example: Network Container A is non-IB-trackable. It contains two items: Item 1 and Item 2. Item 1 is non-IB-trackable, and Item 2 is IB-trackable. In the search

results, Network Container A is displayed along with its items.

Note that customers cannot start the reconfiguration process if any of the instances has a pending order against it. In this case, an error message is displayed.

Customers are prevented from disconnecting all services online. At least one service must remain active.

For more information on the My Products page, see the *Oracle iStore Implementation and Administration Guide* (Implementing Carts and Orders chapter).

Using Oracle HTML Quoting

Topics in this section cover working with the HTML version of Oracle Quoting to create and maintain quotes using the Oracle TSO solution.

For information on using the Oracle TSO solution using the Forms version of Oracle Quoting, see the section, "Using Oracle Forms Quoting", later in this chapter.

For detailed information about working with quotes, see the *Oracle Quoting User Guide*.

Configure New Instance

When a sales agent creates a new configuration for his customer, he can either create a new quote or use a quote from a template. In either case, following are the basic steps:

- Specify a name for the quote.
- Specify the customer's name and location by performing a search.
- Change default values, if necessary.
- Select the appropriate order type that allows the process to flow into Oracle Service Fulfillment Manager.

For more specific information about creating a quote, see the *Oracle Quoting User Guide*.

Add Products to the Quote

To add products to a quote, in the Products tab, the sales agent selects the Add Product button. Following are additional guidelines specific to TSO functionality:

- When searching for products, if a product is included (contained) in a container model, the sales agent must search by the product name of the container. If he searches by a product name that is in the container, he cannot place those products into the quote.

Add Telecommunication Services

To add telecommunications services to a quote from the customer's installed base, in the Products tab, the sales agent selects the Add Product button and then chooses Installed Base as the Search In source on the Search and Select: Product page. Following are additional guidelines specific to TSO functionality:

- Installed base searches can only be performed on root trackable items. The search results show only root trackable items.
- As a prerequisite, there must be existing TSO products in the customer's installed base.
- From the Action menu of the Search and Select: Product page, the sales agent should ensure that Reconfigure is selected.
- In the Search and Select: Product page, the agent can search for products by various criteria, such as Serial Number, Installed At City, Instance Number, and Instance Name.

Reconfigure Telecommunications Service

To reconfigure a customer's existing telecommunications service, the sales agent retrieves the Products page and selects the network container for the telecommunications service. Following are additional guidelines specific to TSO functionality:

- The product to reconfigure must be a telecommunications service.
- From the Select Line list-of-values, the agent chooses Configure and clicks Go to retrieve Oracle Configurator. The agent reconfigures the item as necessary. After the agent closes Oracle Configurator, the Products page appears with the new configuration.
- Optionally, the agent may expand the network container to see the components. The Action field indicates any action relative to the component during reconfiguration. Oracle Configurator returns this line type, which appears in the Actions column, to Oracle Quoting. If the configuration is complete, the total price of the model appears following the description.
- For more information on using Configurator, see *Oracle Configurator Methodologies*.
- Agents cannot perform the following actions on a telecommunication service from Oracle Install Base: split line, remove changed lines, or add services.

Add to Container Model

To add more items from Oracle Installed Base to the container when there exists a container model in a quote with items from Installed Base, the agent selects the container to which he wishes to add a product; from the Action menu, he chooses Add to Model From Installed Base. Following are additional guidelines specific to TSO functionality:

- A search of a customer's installed base could result in multiple search result pages. Only installed products from one search results page can be added to a quote. To select products from additional search result pages, the agent can use the Add to Model from Installed Base action.
- The search for items is limited to items in the selected container model.
- Installed base searches can only be performed on root trackable items. The search results show only root trackable items.

Remove Unchanged Components

To remove unchanged Components of a configuration, from the Products tab, the agent selects Quote Information to retrieve the Lines page. In the Lines page, select Remove Unchanged Products. Following are additional guidelines specific to TSO functionality:

- An unchanged component is one that either:
  - The user explicitly selected from Oracle Installed Base, brought the component into a configuration session, but did not change it; or
  - A passive component that was made active during the configuration session, but did not change.
- If after reconfiguring the telecommunications service, the agent wishes to view only the changed components, he can choose to remove all unchanged components. This is helpful to see the difference in cost after reconfiguration. Each unchanged component has a visual indicator in the Actions column, alerting you that the component is unchanged during the configuration session.
- After the agent removes unchanged components in the quote, he cannot add the unchanged components to the quote by re-launching Configurator. An unchanged component, when removed from the quote, continues to be visible in Configurator. The visual indicator is a triangle that contains an exclamation point.
- As a prerequisite, a line in the quote must be unchanged from the original configuration.
- You cannot undo the Remove Unchanged Components action. Before performing

this action, ensure that this is the action that you want to perform.

Remove Lines

To remove unchanged components individually, or remove entire container models, from the Quotes tab, select Quote Information to retrieve the Lines page and select the line to remove. Following are additional guidelines specific to TSO functionality:

- The line to remove must not be a changed Component of the Container Model.
- From the Select Lines menu, choose Remove Line and press Go.

View Payment Due with Order Amount

A Payment Due with Order payment is an amount that the customer is required to pay the service provider at the time of placing an order. It is usually a portion of the Total Price. This also is known as the Payment Due with Order amount. For example, in a sale of TSO services and equipment, the charges for the equipment (e.g., a cell phone), service activation, shipping and handling, and taxes could be pay now payments. In Oracle Quoting, the Payment Due with Order amount is displayed in the product line in the Products tab, Quote totals, and in the Print Quote report. The Payment Due with Order amount is determined by the payment terms at the quote header and line levels. The amount is the sum of all the due with order amounts such as the price, tax, and pricing charges.

Also note the following:

- The Due with Order function must be enabled.
- The Due with Order column is hidden by default. You must enable it for display using OA Personalization.
- At the line level, the Due with Order column appears after the Total Charges column by default.
- In the Quote Totals region, the Due with Order details appear after the Recurring Charges details.
- In the Print Quote report, the Due with Order subtotal appears after all the other subtotals in the quote.

For more information, refer the *Oracle Quoting Implementation Guide*.

View Recurring Charges

Sales agents in Oracle Quoting can view the charge periodicity for each quote line in the quote and in quote templates. They also can view the quote totals and the breakdown

for the recurring and one-time charges.

In the Quoting Products page, the column, Charge Periodicity, displays the charge periodicity associated with each line item in the quote. Items without a charge periodicity show nothing in this column. For items that have recurring charges associated with them, the List Price column of the Products page shows the recurring charge amount.

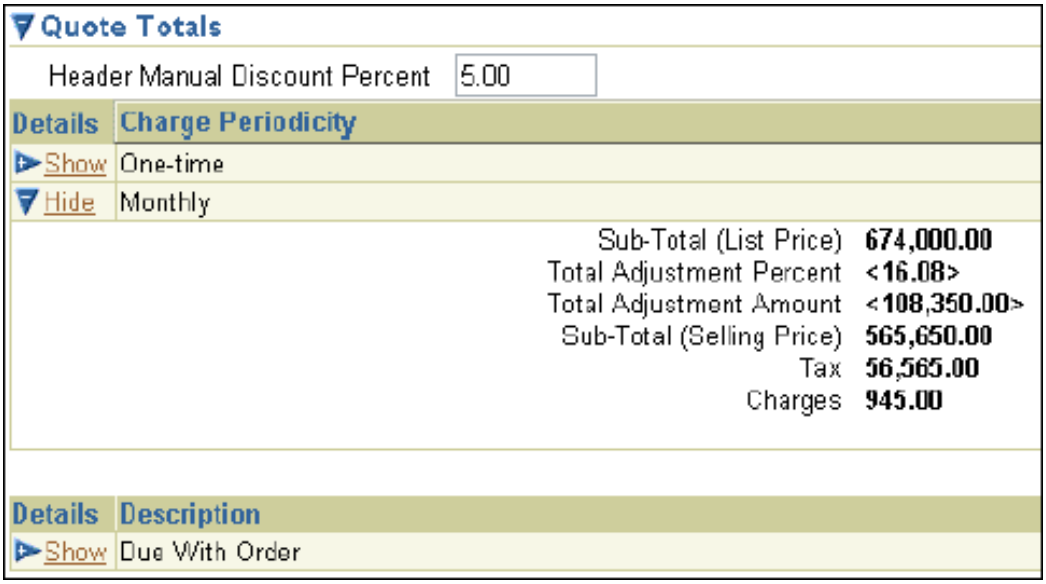
The Update Template and Template Details pages also display the Charge Periodicity column.

After being exposed by the implementer as a setup step, the Quote totals area displays both recurring and one-time charges in separate sections that feature show/hide icons. The recurring charges area shows:

- Sub-Total (List Price): This is the sum of (List Price times Quantity) for each quote line for the periodicity.
- Total Adjustment Percent: This is the Total Adjustment Percent applied to the quote lines.
- Total Adjustment Amount: This is adjustment the total amount applied to the quote lines for the periodicity.
- Sub-Total (Selling Price): This is List Price minus Total Adjustment Amount.
- Charges: If charge periodicity is null (meaning there are one-time charges), this will be the sum of header-level pricing charges and quote line-level pricing charges where the periodicity is null. If periodicity is not null (meaning there are recurring charges), this will be the sum of the quote line-level pricing charges for the periodicity.

The following graphic shows an example of how recurring charges display in HTML Quoting.

View Recurring Charges in HTML Quoting



A Total column displays to the right of the one-time and recurring charges. For each line, the Total column is the quote total for the quote lines having a particular periodicity. Header level pricing charges are included in the one-time total. The computation is: Sub-Total (Selling Price) + Tax + Charges.

Place an Order

To convert a quote into an order, from the Quotes tab, the agent selects the Overview link to retrieve the Quote Overview page; he then selects Place Order from the Select Line menu. Following are additional guidelines specific to TSO functionality:

- When the agent converts the quote into an order, Oracle Service Fulfillment Manager (if integrated) retrieves the order from Oracle Order Management. Service Fulfillment Manager fulfills the order, and the service changes appear in Oracle Install Base.

- Several prerequisites must be met, as described in the *Oracle Quoting User Guide*.

Using Oracle Forms Quoting

Topics in this section cover working with the Forms version of Oracle Quoting to create and maintain orders using the Oracle TSO solution.

Configure a New Instance in Forms Quoting

To configure a new instance, the sales agent navigates to Forms Quoting and creates a

new quote. For more information, refer to the *Oracle Quoting User Guide*.

Add Products the Quote

To add products to a quote, the sales agent navigates to Quoting and selects the Quote Lines tab. Following are additional guidelines specific to TSO functionality:

- As a prerequisite, the customer must appear on the Header tab of the Oracle Telecommunications Service Ordering form.
- The agent should select the first available line of the Product field. If the Product field already displays a product, the agent should select New Record on the toolbar.
- The agent may choose a product from the Product list, or enter the product description or part number. The functioning of this field depends on whether implementers have enabled Oracle interMedia Search (see the section on using Oracle interMedia Search in the in the *Oracle Quoting User Guide*).
- Alternately, the agent can perform a product search by choosing Actions, Product Search to copy and paste the part number from the search results to the Product field.
- To view the product details, the sales agent selects Product Details. The Product Details form shows the item type and whether the product is returnable, is serviceable, is shippable, or is a support service.
- Optionally, the sales agent may update the default UOM value.
- The Quantity field is updateable. The ASO: Default Ordered Qty in OC UI profile option provides the default quantity.
- From the Price List, the sales agent chooses a price list for the line. He can only select a price list if the ASO: Price List Override profile option is set to Yes.
- If the item type is Model, then the product is configurable. The displayed product is the base model. The agent can configure this product to add more components. For more information, see the section on configuring a product in the *Oracle Quoting User Guide*.
- For all other products, the product pricing appears. In the Price List field, the agent can view the unit price.

Reconfigure Telecommunications Service

To reconfigure a telecommunications service, the sales agent navigates to the Quotes form and creates a quote. Following are additional guidelines specific to TSO functionality:

- The sales agent may add products to the quote, and then search the install base for the telecommunications services to reconfigure; he reconfigures them by launching Oracle Configurator.

- The agent cannot perform the following actions on a telecommunication service from the install base: split line, duplicate lines, add services, or designate line level shipping or billing.

Add Telecommunications Services

To add telecommunications services for reconfiguration, the sales agent selects Change Installed Base. Following are additional guidelines specific to TSO functionality:

- As a prerequisite, there must be existing telecommunication services in the customer's install base.

- The agent may search by:
  - Category
  - Product
  - Description
  - Order Number
  - Order Date (From)
  - Order Date (To)
  - Instance Name
  - Instance Number
  - System
  - Serial Number
  - Installed At City
  - Installed At Country
  - Attribute Name
  - Attributes Value

The search results include:

- Instance ID
  - Root Instance ID
  - Product
  - Description
  - UOM
  - Quantity
  - System Name
  - Serial Number
  - Installed At Address
- Relationships of the telecommunication services appear in the Components/Connections bin.
- If the Attributes bin is available, the agent can view the attributes for each telecommunications service.
- Optionally, the agent may view details for the related telecommunication service by selecting the service and clicking Details.
- After the agents adds the service to the quote, a network container that includes the telecommunications services appears in the quote. If the configuration is complete, the total price of the model appears in the Line Total field.
- The configuration appears as a new line in the Pricing region of the Quote Lines tab. The Line Type field indicates any action relative to the component during reconfiguration. Oracle Configurator returns this line type to Oracle Quoting.
- If there are no changes on a line, a visual indicator appears in the Line Type field to alert the agent that there was no change to the original configuration.
- The quantity and UOM for the network container and telecommunication service default into Oracle Quoting from the installed base.
- The network container quantity must always be 1 and should not be changed.
- The network container price list defaults from the header level price list on the quote. The model and all components also have the default line category ORDER.

Reconfigure a Service

To reconfigure telecommunication services, the agent selects Actions, Add to Installed Base Changes, OK. Following are additional guidelines specific to TSO functionality:

- As prerequisites, the product to configure must be a telecommunications service.
- ASO: Use Network Container must be set to Yes.
- When searching, search results are restricted to telecommunications services related to the model in the quote. Relationships of the telecommunication services appear in the Components/Connections bin.
- Optionally, the agent may view details for the related telecommunication service by highlighting the service and selecting Details.
- After closing Oracle Configurator, the Quote Lines tab displays the new configuration. The Line Type field indicates any action relative to the component during reconfiguration. Oracle Configurator returns this line type to Oracle Quoting. If the configuration is complete, the total price of the model appears in the Line Total field.

Remove Unchanged Components

To remove unchanged Components of a configuration, the agent selects the Quote Lines tab; he then selects Actions, Remove Unchanged Products, OK. Following are additional guidelines specific to TSO functionality:

- If after reconfiguring a telecommunications service, the agent wishes to view only the changes, he can choose to remove all unchanged components. Viewing only changed components is helpful if the agent wishes to see the difference in cost after reconfiguration. Unchanged components include an asterisk in the Line Type field.
- After the system removes unchanged components, the pricing for the telecommunications service reflects only the changed components.

Remove Lines

To remove products from the quote, the sales agent selects the Quote Lines tab; he then selects the line to remove and chooses Delete from the Edit menu.

- As a prerequisite, at least one product must be in the quote.
- After the system removes the line from the quote, it renumbers the remaining lines.

View Payment Due with Order Amount

A Payment Due with Order payment is an amount that the customer is required to pay the service provider at the time of placing an order. It is usually a portion of the Total Price. This also is known as the Payment Due with Order amount. For example, in a sale of TSO services and equipment, the charges for the equipment (e.g., a cell phone), service activation, shipping and handling, and taxes could be pay now payments. In Oracle Quoting, the Due with Order payment is displayed in the Quote Header-Summary region, Quote Line - Pricing region, Quote totals, and in the Print Quote report. The Due with Order amount is determined by the payment terms at the quote header and line levels.

In addition, note the following:

- The Due with Order function must be enabled.
- The Due with Order column in the Quote Line - Pricing region is hidden by default. You must enable it for display. The amount is the sum of all the due with order amounts such as the price, tax, and pricing charges.
- To view the Due with Order details for a quote, the agent should choose the View Due with Order Details action from the Header level Actions form.

For more information, refer to the *Oracle Quoting Implementation Guide*

Place the Order

To convert a quote into an order, in the Quote Overview page, the agent selects Place Order from the Actions menu.

- After the quote is converted into an order, Oracle Service Fulfillment Manager (if integrated) retrieves the order from Oracle Order Management. Service Fulfillment Manager fulfills the order, and the system updates the customer's install base with the service changes.
- Several prerequisites must be met, as described in the *Oracle Quoting User Guide*.

Using Oracle Order Management

This section contains information on using TSO-specific aspects of Oracle Order Management.

Payments Window - Header Level

In the header-level Payments window, Oracle Order Management provides total payment amounts, and the total due now (Payment Due with Order) amounts for the order. Following are additional points about this window:

- Outbound Subtotal: This is all charges due on the order, including taxes.
- Total: This is total payments, including initial due subtotal, taxes, all charges, any prepaid amount, commitments, and balance (does not include credit for RMA lines).
- If the system parameter, Installment Options, is set to Enable Pay Now, the subtotal, charges, and tax display are only for the Payment Due with Order amount portion of all charges on the order. Recurring charge lines should have pay later payment terms; otherwise, the pay now amount portion of these charges will be included in the Total Due Now amount. In this case, the checkbox, Due with Order, is checked.
- If the system parameter, Installment Options, is set to Authorize First Installment, the subtotal, charges, and tax displayed are only for the first installment of all charges on the order. In this case, the checkbox, Due with Order, is not checked.
- If the system parameter, Installment Options, is set to None, no amounts are displayed in the fields, and the fields are not visible to add to the folder block.
- You can create payment records for the balance due.

Note: The Authorized Amount on the Payments window is read only.

Payments Window - Line Level

The line-level Payments window can only be invoked for outbound lines. Whether the amount on the line is pay now amount or pay later is determined by the payment term. The Line Payment window displays the following:

- The subtotal, tax, charges, total, commitment, and balance for the total amount for the line and the initial due total for the line
- If the system parameter, Installment Options, is set to Enable Pay Now Amount, the subtotal, charges, and tax displayed in the Initial Due fields are only for the Payment Due with Order portion of the charges on the line. In this case, the checkbox, Due with Order, is checked.
- If the system parameter, Installment Options, is set to Authorize First Installment, the subtotal, charges, and tax displayed in the Initial Due fields are only for the first installment of the charges on the line. In this case, the checkbox, Due with Order, is not checked.
- If the system parameter, Installment Options, is set to None, no amounts are displayed in the fields, and the fields will not be visible to add to the folder block.

Note: Multiple credit card authorizations are possible, so only the latest authorized amount displays.

Authorizing Credit Card for Payment Due with Order Amount

Note the following about authorizing credit cards for the Payment Due with Order amount:

- Authorization is done once for the credit card specified and for the Payment Due with Order amount total on the order. Authorization is not done per line.
- The column authorized amount in the Line Payments region in the Payments window (header level) is updated with the authorized amount.
- The payment term on the header is interfaced to Oracle Receivables for the header-level charge, not the payment term of the line. This includes the tax amount in the Payment Due with Order amount portion.

Interfacing Freight Amounts

The Oracle Order Management Invoice Interface module interfaces charges in two different ways to Oracle Receivables depending on the following profile options:

- TAX: invoice freight as revenue
- TAX: allow override of tax code

If both of these profile options are Yes, charges are interfaces with Line Type = LINE; otherwise they are interfaced with Line Type = FREIGHT.

Note: If tax is inclusive in the lines amount, the tax calculation for Payment Due with Order amount portion may be incorrect.

Modifying Payment Due with Order Terms

You can change a line from pay now to pay later after booking (after authorization), or you can set up a processing constraint to prevent this change from occurring. You also can change the installment information for a payment term so that it is no longer a Payment Due with Order amount payment term. Also note the following:

- Use the Payments window to capture payment information for the Payment Due with Order amount total.
- You can enter information only once for the Payment Due with Order amount total.
- You may specify only one payment type for the header-level payment of an order. This payment type applies to both Payment Due with Order amount and pay later amount lines.
- You can specify different payment instruments at the Payment Due with Order amount lines.

- You may specify one payment type for line-level payment on each order line (in addition to commitment).
- You can apply the pre-payment toward the Payment Due with Order amount total.
- You may specify one or more payment types for the pre-payment.
- You cannot use line-level payments with pre-payment.
- Oracle Order Management authorizes a credit card only once for the Payment Due with Order total. Amount to authorize is equal to Payment Due with Order total, minus pre-payment, minus commitment (of the Payment Due with Order lines). The pay later amount is not included in the amount to authorize.
- Changing the payment term after credit card authorization may result in Over authorization or Under authorization.
- Different payment methods for Payment Due with Order amount and pay later amount are not supported in Oracle Order Management.

Oracle Receivables and Payment Due with Order Totals

Note the following about Oracle Receivables and Payment Due with Order totals:

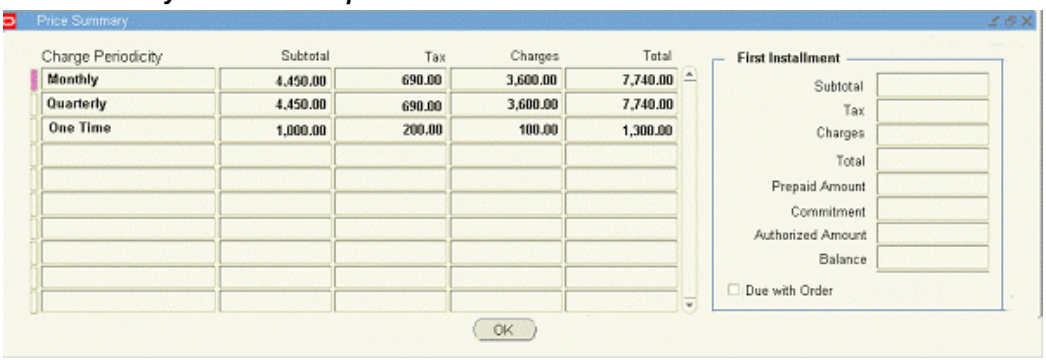
- The Payment Due with Order amount is calculated by multiplying the line amount by the percent due of any or all installments due in 0 (zero) days, specified on the payment term. Taxes and charges are calculated according to the payment term definition.
- If the optional balance forward billing is used, all lines interfaced to Oracle Receivables can have distinct payment terms, resulting in multiple invoices. All of the individual invoices are aggregated and included as line items on a single, balance-forward invoice. Payment terms on the individual invoices are overwritten by a single, customer-level payment term when the balance forward bill is generated.
- For Payment Due with Order amounts with a credit card, ACH, or direct debit payment type, Oracle Receivables captures funds on the due date of the individual invoices associated with the Payment Due with Order amount total (Since payment term is immediate for these lines, the payment is captured upon invoicing via a concurrent request that is scheduled immediately after AutoInvoice).
- Oracle Receivables bills the customer account on a specific billing cycle for the pay later total through balance forward billing (consolidated billing).

Note: When an order line has a Payment Due with Order amount portion, the entire line is considered a Payment Due with Order amount line.

Price Summary Window

Payment Due with Order amounts display in the Price Summary window. The following graphic shows an example of the Price Summary window displaying Payment Due with Order amounts.

Price Summary Window Example



Sales Order/Quick Sales Order Windows

The Sales Order and Quick Sales Order windows display the Payment Due with Order amount (the Balance Due total amount). An action in the Sales Orders and Quick Sales Orders windows opens the Price Summary window that displays the Payment Due with Order amount and any recurring charges on the order.

Note: The value for the Balance Due will only be populated if the Oracle Order Management system parameter, Installment Options, is set to either Enable Pay Now or Authorize First Installment.

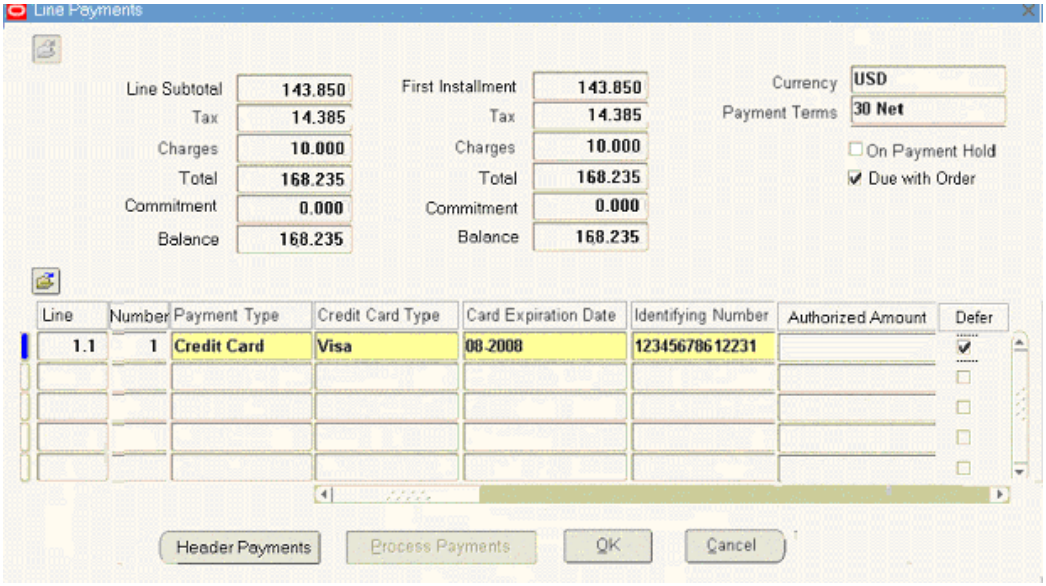
The recurring charges totals can be seen on the order header. The order header window provides a column named Charge Periodicity with an LOV consisting of all types of charges applicable in the order. User can change the non-updatable column to see the charge totals for the charges applicable to that order.

The field, Adjusted Amount, in the View Adjustments window only contains the one-time amount. The button, Recurring Amounts, allow the display of recurring amounts per charge periodicity for the order-level adjustment. This button is enabled only when the system parameter, Enable Recurring Charges, is Yes.

View Line Payments Window

When the Line Payments window is invoked from the Payments window, all payments for the order lines are displayed in the multi-record block that shows payments. The Line Payments window displays the total pay now amount and the total balance due amount. The following graphic shows an example of the View Line Payments window

View Line Payments Window Example



Place Order and View Payment Due with Order Amount

To place an order and view the Payment Due with Order amount:

1. Create a sales order header with the customer information.
2. Enter the information for the order lines.
3. Choose Actions, then Payments to open the Payments window.
4. View the Balance under First Installment to see the Payment Due with Order amount portion.
5. You can now tell the customer the Payment Due with Order amount total that will be authorized when the order is placed.
6. Create a payment record. Enter the payment information (such as payment type code, credit card number, credit card holder name, etc.) that is used to pay for the Payment Due with Order amount. Book the order, and if authorization fails, the order is booked and placed on hold.
7. Print the payment receipt to show the order and payment tendered.
8. The Payment Assurance workflow activity ensures that a payment record exists for the Payment Due with Order amount total before fulfilling the order.
9. Fulfill the order line.

10. The Payment Due with Order amount information is interfaced to Oracle Receivables for invoice creation.

Payment Due with Order Amounts in Oracle Order Management Reports

The Oracle Order Management Daily Business Intelligence reports, Comprehensive Order Detail and Payment Receipt, display Payment Due with Order amounts as follows:

Comprehensive Order Detail Report

- Initial Due Total
- Initial Due Balance
- Authorized Amount

The above amounts display for both the order header and line items.

Payment Receipt Report

- Initial Due Total
- Payment Tendered
- Commitment Applied
- Authorized Amount

TSO-Related Order Management Workflows

Several workflows and workflow activities are relevant to the TSO solution in Oracle Order Management.

Payment Assurance Workflow Activity

This workflow activity ensures that the Payment Due with Order amount has been recorded and authorized (for credit card payment), when Payment Due with Order functionality is enabled.

Note: You must place this workflow activity before fulfillment.

Line Flow – Generic, Bill Only With Payment Assurance Workflow

This seeded workflow holds the Payment Assurance workflow activity before the Fulfill activity to guarantee that the payment has been ensured before the product/service on the order line is fulfilled.

Invoice Interface Workflow Activity

This workflow activity can be used in an order line workflow to interface payment

information for an order line. This activity handles the Payment Due with Order functionality. If Payment Due with Order is enabled, the activity will:

- For the Payment Due with Order amount line, interface the credit card authorization code
- For the pay later amount line, interface payment information excluding authorization code

Note: Interface header level charge with Line type = Freight, and payment term ID is taken from the order header (not the order line) if Payment Due with Order is enabled.

Line Flow – Generic, with Payment Assurance

This seeded workflow holds the Payment Assurance workflow activity before the Schedule activity to guarantee that the payment has been assured before the product/service on the order line is scheduled.

17

Administering the Oracle TSO Solution

This chapter covers the following topics:

- Overview of Administering the Oracle TSO Solution Chapter
- Re-Validate Installed Configurations After Modifications
- Managing Notification Errors
- Find an Order in SFM Order Flow-Through Area
- Retry Install Base Updates
- Retry Failed Outgoing Messages
- Manage Failure Notifications in HTML

Overview of Administering the Oracle TSO Solution Chapter

This chapter provides information about adminstering the Oracle Telecommunications Ordering (TSO) solution after setup.

Re-Validate Installed Configurations After Modifications

If you modify a container model in either Oracle Applications or Oracle Configurator Developer, model data can become out of synchronization with configured components at a customer's site. In this case, it is strongly recommended that you re-validate any installed configurations.

The following are examples of changes that occur in Oracle Applications to a container model that require you to re-validate installed configurations:

- Deleting a trackable child model
- Making a trackable child model non-trackable

The following are examples of changes made in Oracle Configurator Developer that require you to re-validate a container model:

- Deleting a connector node from a trackable model, when a rule contains participants from the connector's parent and the target model.
- Adding, deleting, or changing a rule defined using a connector, when the rule contains participants from two trackable components.
- Adding, deleting, or changing a rule in a trackable model that is not defined using a connector, but the rule propagates to a participant in a rule defined using a connector between trackable components.
- Adding, deleting, or changing any item that is a participant in a rule defined using a connector, when the connector's parent and target are both trackable.

In this procedure, log in to Oracle Configurator Developer with Configurator Administrator or Configurator Developer responsibility.

Steps

1. Open the container model.
2. Regenerate the active model.
3. Republish the model.
4. In Oracle Quoting, create a new quote.
5. Search for all components within the customer's existing network configuration.
6. Select all of the components, and add them to the quote. If this step fails due to a large number of components, create multiple quotes with fewer sets of connected components, then continue the validation process separately for each quote. For each quote, try to select sets that have very few, if any, connections between each set.
7. If the changes made to the container model do not affect the customer's installed configurations, verify that all items appear with your No Action or Reprice Line Type. Because you define Line Types in Oracle Order Management, the exact text can vary. For more information, see the *Oracle Order Management Suite Implementation Manual*.
8. Verify that any items affected by the changes made in the container model are invalid within the configuration.
9. Submit the quote. If no errors occur, you have re-validated the configuration successfully.
10. If errors occur:

1. Launch Oracle Configurator.
2. Update the configuration to resolve each error.
3. Resubmit the quote.
4. Repeat steps a-c until you can submit the quote with no errors.

Managing Notification Errors

Enhancements to Oracle Service Fulfillment Manager (SFM) that assist you with notification errors while using the Oracle TSO solution include:

- Incorporating enhancements to allow you to act upon notifications created as a result of a failed action
- Improving interactions with third parties by automatically re-sending XML messages to them if the original XML message failed to send

Notification error-related tasks include:

- "Retry Install Base Updates"
- "Retry Failed Outgoing Messages"
- "Manage Failure Notifications in HTML"

Find an Order in SFM Order Flow-Through Area

You should monitor your sales orders to check for error notifications.

In this procedure, log in to Oracle Forms with SFM System Administrator responsibility and navigate to Operations, Flow-Through Manager.

Steps

1. In Order Num field:
   - Specify the OM Order Number.
   - Choose Sales Order as Source Application in the Search window.
   - Click Go.
2. Expand the order in the tree to view the Service Fulfillment lines that Oracle Service Fulfillment Manager has received.
3. Choose a line and navigate to the Work Items tab.

4. Click Parameters.

5. View the parameters that the system passed from the line into Oracle Service Fulfillment Manager.

Retry Install Base Updates

Oracle Service Fulfillment Manager (SFM) updates the install base with provisioning data after an order is complete in SFM. If the install base update fails, SFM sends a response notification to the SFM System Administrator. The System Administrator has the option to retry the failed action.

It is recommended that you routinely monitor sales orders to be aware of errors that can occur, such as when provisioning fails. Provisioning failure can occur when there is an incompatible discrepancy between the order and the information in either Oracle Service Fulfillment Manager or the install base. A provisioning failure appears on the Sales Order form in the status of the flow.

After you have fixed any problems pertaining to the incompatible discrepancy, you can retry to update the install base.

In this procedure, log in to Oracle Forms with SFM System Administrator responsibility and navigate to Operations, Flow-Through Manager.

Steps

1. Enter search criteria to find the Order Number.

2. Select the Order Number, and click View Details. The Order Information window appears.

3. Click the Notification button.

4. The Notification Inbox opens.

5. Select a notification whose status is Open.

6. Click View Details.

For more information on your options, such as retrying to update the install base, refer to the *Oracle Service Fulfillment Manager Implementation Guide.*

Retry Failed Outgoing Messages

The XDP Resubmit Failed Message concurrent program resends XML messages that have failed to be sent successfully to third-party systems. The concurrent program takes a message code as input. When you run the concurrent program, SFM identifies and attempts to resend all failed messages with the specified message code.

You should run the XDP Resubmit Failed Message program if:

- You have information that some of your messages failed to send
- You want to monitor and verify the sending of your messages.

In this procedure, log in to Oracle Forms with SFM System Administrator responsibility and navigate to Concurrent, Run Requests.

Prerequisite

You need to know the Message Code of the message that you want to check.

Steps

1. Select Single Request and click OK.
2. In the Name field, choose XDP Resubmit Failed Message.
3. Click OK.
4. Enter the Message Code.
5. Click OK. Any messages that the system had failed to send reappear in the Outbound Message Queue. An adapter program polls the queue to send messages from the queue.
6. To view messages in the Outbound Message Queue, choose Administration, Queue Console.
7. In the Queue Console, select and open the Outbound Message Queue. The Entries field shows the number of messages in the queue.
8. Select a message and click the View Details button.

Manage Failure Notifications in HTML

SFM includes an HTML version of the Fallout Manager that you access to manage failure notifications. You view the automatically created failure notifications in HTML using Oracle Workflow. From the Workflow Notification page, you can navigate to the SFM HTML user interface to modify the workitem parameters and retry the failed action.

In this procedure, log in to Oracle Forms with Workflow Administrator Web (New) responsibility and navigate to Status Monitor area.

Prerequisite

See "Add Function for Menu Notifications" in the chapter, Set Up Service Fulfillment Manager.

Steps

1. In the Status Monitor area, click the Notifications link.
2. The Workflow Notifications page opens and displays: Type, Subject, and notification Date.
3. To view details of the order number notification, click the Subject hypertext link. The Notification Details page shows the selected failure notification message and hypertext links that let you fix the errors, such as Workitem Parameters.
4. In the Details area, click the link to update Workitem Parameters. The Workitem Parameters for Failed Order Line page opens.
5. In the Retry Value column, enter corrections to the workitem parameters.
6. Click the Update button. The Workitem Parameters for Failed Order Line page closes and the Notification Message Details page opens.
7. Click the Retry button.

A

Assumptions and Restrictions

This appendix covers the following topics:

- Overview of Assumptions and Restrictions Appendix
- General Assumptions
- TSO With Equipment Assumptions
- Payment Due With Order Assumptions
- Recurring Charges Assumptions
- Oracle Configurator Integration Assumptions
- Oracle Inventory Integration Assumptions
- Oracle Order Management Integration Assumptions
- Oracle Quoting Integration Assumptions
- Oracle Service Fulfillment Manager Integration Assumptions

Overview of Assumptions and Restrictions Appendix

This chapter lists assumptions and restrictions for Oracle Telecommunications Service Ordering (TSO) functionality.

General Assumptions

Following are general assumptions about the implementation and use of the Oracle TSO solution:

- Users cannot make changes to pending orders. The following are not supported:
  - Placing new quotes based on pending orders
  - Modification of entered but non-booked orders by launching Configurator from

the Order Management Sales Order Pad (users can modify the data on booked orders, but they cannot reconfigure them once the orders are booked)

- Users can reconfigure order lines for an order that is in the Negotiation phase. The line workflow for an order line of an order that is in the Negotiation phase does not start until it becomes an order in the Entered state.
- Users cannot reconfigure order lines that are part of a container model after booking the order.
- Users can reconfigure order lines for an order that is in the Entered state if the workflows for the line types are the same. When users enter an order line, the workflow for the order line starts. For example, if a user reconfigures an order line such that its line type changes from Disconnect to Change, the workflow that implements the line types Disconnect and Change must be the same.
- Orders that hold container models are solution-based models. Importing of solution-based models is not supported.
- From the order, users cannot directly remove an order line that is holding an item instance that has been reconfigured in Configurator. To remove such an order line, users must: invoke Configurator, undo the change, and then explicitly remove the order line.
- Users cannot copy order lines that hold reconfigured item instances.
- Users cannot copy a booked order that is holding reconfigured item instances.
- Orders that hold container models require transaction details to hold, for example, item instance attribute values and relationships. Importing of transaction details as part of an order import is not supported.
- Users can cancel an order (or all order lines) belonging to the same model. To prevent cancellation of orders, implementers can set up processing constraints.
- It is not possible to convert an order in the negotiation phase into an order if the revision of the item instance in the installed base is higher than the revision of the item instance on the order line of the order that is in the negotiation phase.
- The line type for an order line that is holding a component that is part of a container model cannot be changed on the order line itself; the line type for such a line can only be changed by invoking Configurator. Configurator will feed the appropriate line type back to the order line based on the rules set up in Configurator.
- Users can only reconfigure instances of quotes and orders for the sold-to customer who owns the service in Oracle Installed Base. For example, Customer X cannot reconfigure services belonging to Customer Y.

TSO With Equipment Assumptions

Following are the assumptions with respect to equipment in the Oracle TSO solution:

- Equipment items are allowed in Order Management/Quoting only in Add and Disconnect flows.
- Only the shipping of standard items and kits within a container is supported. Shippable PTO models or option classes, or any ATO models, option classes and items are not supported.
- The quantity of the trackable root model (MI, PTO model) will always be 1 and will be non-shippable.
- Shippable and inventory transactable items cannot have a quantity greater than 1.
- Removal of shippable equipment from a configuration will result in deleting the item instance in Configurator and decoupling the instance from the configuration in the installed base. In order to return the item removed from the configuration, the standard RMA process needs to be followed for the decoupled instance.
- A replacement flow, as understood by IB Replace_by relationship (i.e., transfer of warranties, etc.) is not in supported.
- Using Oracle Installed Base extended attributes for shippable items in Oracle Shipping is not supported. This applies to items captured in Configurator.
- If an item is flagged as Shippable, then it should also be flagged as Inventory Transactable and vice versa.
- In-store equipment exchange or loaner is not supported.
- No connectors to shippable items are allowed. This is because connectors can only be defined against PTO Models or ATO Models and those cannot be shippable. This requirement is avoids failing installed base transaction relationships pointing to already returned instances.
- If a shippable item is ordered but not shipped (just handed to the customer by the reseller for instance), there is no way to capture the serial number of the new item.
- Oracle Depot Repair integration is not supported.
- The TSO solution is dependent on Instance Locking functionality (in Configurator, Installed Base, and Order Management).
- The Container Model is not serviceable, since it is an artificial, non-trackable item on the order.

- A tangible item is a serial controlled item as defined in Oracle Inventory

Payment Due With Order Assumptions

Following are the assumptions with respect to Payment Due with Order (Pay Now) amounts in the Oracle TSO solution:

- The Payment Due with Order amount is charged through Oracle Receivables, not a third party billing system.
- If a line has a Payment Due with Order component, the entire line is handled through Oracle Receivables.
- If an order has all (only) recurring charge lines and a header level charge, the header-level charge is sent to Oracle Receivables.
- Recurring charges are not included in the Payment Due with Order total.
- The payment method for the Payment Due with Order amount is not programmatically restricted to credit cards only.
- Deposits are not supported.

Recurring Charges Assumptions

Following are the assumptions with respect to recurring charges in the Oracle TSO solution:

- All of the selling applications use the same mechanism to default periodicity.
- Oracle Order Management does not over-write the periodicity values from Oracle iStore and Oracle Quoting.

Oracle Configurator Integration Assumptions

Following are Oracle Configurator integration assumptions about the implementation and use of the Oracle TSO solution:

- To successfully reconfigure an instance, implementers must accurately follow all setup steps and tips.
- Implementers should not connect two or more connectors from one source item to one target item.
- It is recommended that users do not use pricing in the Oracle Configurator

Summary window. It is recommended that all pricing of TSO products should occur through the hosting applications.

- Configurator cannot infer disassembly sequences from the delta computations. A consultant must build the fulfillment workflows.
- Certain types of substantial changes to a configuration model require the developer to re-validate existing instances. These changes include:
  - Changing a trackable, multiple instantiable child Model of the container model to non-trackable
  - Deleting a trackable, multiple instantiable child Model of the container model
  - Deleting a Connector node from one trackable model if there are connection rules with participants across the Connector
  - Adding, deleting, or changing a connection rule between trackable items
  - Adding, deleting, or changing any item that is a participant of a connection rule between trackable items

Oracle Inventory Integration Assumptions

Following are Oracle Inventory integration assumptions about the implementation and use of the Oracle TSO solution:

- The quantity of an IB Trackable item in a container model is restricted to one.

Oracle Order Management Integration Assumptions

Following are Oracle Order Management integration assumptions about the implementation and use of the Oracle TSO solution:

- Oracle Order Management restricts the quantity to 1 for non-serialized, IB Trackable, container Items. For tangible items, the quantity is also restricted to 1.
- When item instances are selected for reconfiguration from the installed base onto the Sales Order or Quick Sales Order forms, the values for the Bill To Address and Ship To Address for the order lines default from the values stored for these attributes for the item instance in Oracle Installed Base (if available).
- If a customer wishes to add more item instances into a container model that has already been added as an order line in the Sales Order or Quick Sales Order forms, the sales representative positions the cursor on the order line that is holding the container model and selects the function that invokes the Install Base (Item Instance

Query) to select more item instances. When the Item Instance Query form is invoked in this way, the list of available item instances to select in the Install Base are restricted to only show item instances that have been set up in the Oracle Bills of Materials to be a part of the selected container model. In Oracle Quoting, this functionality is represented with a separate action, Add to Model.

- Before you can find item instances from either the Sales Order or Quick Sales Order forms, you must specify the Customer field in the order header.

- In the Sales Order Form or the Quick Sales Order Form, you cannot update the customer information in the order header or the order line when you have selected an item instance on a line.

- TSO functionality is not available from the options window in Oracle Order Management. The TSO functionality is not available if the profile option OM: Use Configurator is N.

- You cannot invoke the window to choose item instances back to order lines from Oracle Install Base after booking the order.

- You cannot update order line attributes: IB Owner, Installed at Location, and Current Location that are holding a reference to an item instance.

- If the function, Remove Unchanged Components, is invoked from the header level in the Sales Order or Quick Sales Order forms, then all the unchanged lines below any container model on the order will be removed. If Remove Unchanged Components is invoked from the line level, only the unchanged lines below the top model or sub-model of the line that the cursor resides on will be removed from the order.

Oracle Quoting Integration Assumptions

Following are Oracle Quoting integration assumptions about the implementation and use of the Oracle TSO solution:

- The function, Remove Unchanged Components, always removes all of the unchanged lines on the quote.

- If a customer wishes to add more item instances into a container model that has already been added as an order line, the sales agent must search Oracle Installed Base to find the item instances. When the Item Instance Query form is invoked in this way, the list of available item instances to select in the installed base are restricted to only show item instances that have been set up in the Oracle Bills of Materials to be a part of the selected container model. This functionality is represented with a separate action, Add to Model.

Oracle Service Fulfillment Manager Integration Assumptions

Following are Oracle Service Fulfillment Manager integration assumptions about the implementation and use of the Oracle TSO solution:

- During fulfillment, Oracle Service Fulfillment Manager can only change user attribute values. No changes occur to structure, quantity, or connection information.
- Fulfillment of orders occur in the same order in which they were booked.

Glossary

Accepted

A quote status indicating that the customer has accepted the quote terms.

API

Application Programming Interface

ATO

Assemble to Order. An environment where you open a final assembly order to assemble items that customers order. Assemble-to-order is also an item attribute that you can apply to standard, Model, and Option Class items. An item you make in response to a customer order.

baseline configuration

The existing configuration, used as the basis for computing a new configuration. Generally, the baseline configuration is the latest booked, provisioned, or installed configuration revision. When updating a configured item, you must reconfigure from this baseline. You can update the baseline to meet your new needs.

bill of material

A list of Component items associated with a parent item and information about how each item relates to the parent item. Oracle Manufacturing supports standard, Model, Option Class, and planning bills. The item information on a bill depends on the item type and bill type. The most common type of bill is a standard bill of material. A standard bill of material lists the Components associated with a product or subassembly. It specifies the required quantity for each Component plus other information to control work in process, material planning, and other Oracle Manufacturing functions. Also known as product structures.

BOM

See bill of material.

BOM item

The node imported into Oracle Configurator Developer that corresponds to an Oracle

Bills of Material item. Can be a BOM Model, BOM Option Class node, or BOM Standard Item node.

BOM Model

A Model that you import from Oracle Bills of Material into Oracle Configurator Developer. When you import a BOM Model, effective dates, ATO rules, and other data are also imported into Configurator Developer. In Configurator Developer, you can extend the structure of the BOM Model, but you cannot modify the BOM Model itself or any of its attributes.

BOM Model node

The imported node in Oracle Configurator Developer that corresponds to a BOM Model created in Oracle Bills of Material.

BOM Option Class node

The imported node in Oracle Configurator Developer that corresponds to a BOM Option Class created in Oracle Bills of Material.

BOM Standard Item node

The imported node in Oracle Configurator Developer that corresponds to a BOM Standard Item created in Oracle Bills of Material.

category

Code used to group items with similar characteristics, such as plastics, metals, or glass items.

CIO

See Oracle Configuration Interface Object (CIO).

Component item

An item associated with a parent item on a bill of material.

concurrent manager

Components of your applications concurrent processing facility that monitor and run time–consuming tasks for you without tying up your terminal. Whenever you submit a request, such as running a report, a concurrent manager does the work for you, letting you perform many tasks simultaneously.

concurrent process

A task in the process of completing. Each time you submit a task, you create a new concurrent process. A concurrent process runs simultaneously with other concurrent processes (and other activities on your computer) to help you complete multiple tasks at once with no interruptions to your terminal.

concurrent processing facility

An Oracle Applications facility that runs time-consuming, non-interactive tasks in the background.

concurrent program

Executable code (usually written in SQL*Plus or Pro*C) that performs the functions of a requested task. Concurrent programs are stored procedures that perform actions such as generating reports and copying data to and from a database.

concurrent request

A user-initiated request issued to the concurrent processing facility to submit a non-interactive task, such as running a report.

configurable item

A base Model item that a user can configure by adding Components.

configuration

A specific set of specifications for a product, resulting from selections that a user made in Oracle Configurator. See also: partial configuration.

configuration attribute

A characteristic of an item that is defined in the host application (outside of its inventory of items), in the Model, or captured during a configuration session. Configuration attributes are inputs from or outputs to the host application at initialization and termination of the configuration session, respectively.

Configuration Interface Object

See Oracle Configuration Interface Object (CIO).

configuration Model

Represents all possible configurations of the available options, and consists of Model structure and rules. It also commonly includes User Interface definitions and Configurator Extensions. A configuration Model is usually accessed in a runtime Oracle Configurator window.

configuration session

The time from launching or invoking to exiting Oracle Configurator, during which end users make selections to configure an orderable product. A configuration session is limited to one configuration Model that is loaded when the session is initialized.

configurator

The part of an application that provides custom configuration capabilities. Commonly,

a window that can be launched from a hosting application so end users can make selections resulting in valid configurations.

Configurator Extension

An extension to the configuration model beyond what you can implement in Oracle Configurator Developer.

A type of configuration rule that associates a Model node, a Java class, and an event binding so that the rule operates when an event occurs during a configuration session.

A Java class that provides methods that can be used to perform configuration actions.

Configurator Extensions have replaced Functional Companions.

connected-to relationships

Models network connections and shows the service configuration. You can view the connected-to relationships of a configured instance from Oracle Install Base. If you define items in Oracle Inventory as link items, you can then use Oracle Install Base to show the start and end locations of the link. The locations for the link instance are the geographic addresses of the instance items.

Connector

The node in the Model structure that enables an end user at runtime to connect the Connector node's parent to a referenced Model.

Container Model

Also called network Container Model. A pick-to-order bill of material (BOM) Model that you import from Oracle Bills of Material into Oracle Configurator Developer which supports multiple instantiation reconfiguration and active and passive Components from Oracle Install Base. This item is not tracked in Install Base.

CTO

Configure to Order

delta

The difference between the new and the baseline configuration.

discontinued item

A discontinued item is one that exists in an installed configuration of a component (as recorded in Oracle Install Base), but has been removed from the instance of the component being reconfigured, either by deletion or by deselection.

Drafted

A quote status indicating that the quote is in the initial phase.

Entered

A quote status indicating that the quote has been successfully submitted as an order in Oracle Order Management. You can modify orders with this status in Oracle Order Management.

equipment

Any tangible item.

fulfillment

Fulfilled sales order lines have successfully completed all Workflow processing activities up to the point of becoming eligible for invoicing.

Functional Companion

An extension to the configuration Model beyond what you can implement in Configurator Developer.

An object associated with a Component that supplies methods that you can use to initialize, validate, and generate customer-centric views and outputs for the configuration.

Functional Companions have been replaced by Configurator Extensions.

ICX

Inter-Cartridge Exchange

installed base/install base

The sum total of all products that a company has responsibility to provide service for at customer sites.

instance

A single representation of an item in a configuration. Actual record of a provisioned product. See also: instantiate.

Also, the memory and processes of a database.

instance name

A method of capturing additional information about a product or instance. An instance name can be either automatically generated or manually assigned during a configuration session.

instantiate

To create an instance of something. Commonly, to create an instance of a Component in the runtime user interface of a configuration Model. See also: multiple instantiation.

item

Anything you make, purchase, or sell, including Components, subassemblies, finished products, supplies, or services. Oracle Manufacturing also uses items to represent planning items that you can forecast, standard lines that you can include on invoices, and Option Classes you can use to group options in Model and Option Class bills.

item attributes

Specific characteristics of an item, such as order cost, item status, revision control, account, and so on.

kit item

A Model defined in Oracle Inventory for the purposes of bundling pre-defined set of items in Oracle Bills Of Material (BOM). At the time of placing an order, a kit does not provide the ability to choose items within it.

location

A point in geographical space that a street address describes. Also, a shorthand name for an address. Location appears in address lists of values to let you choose the correct address based on an intuitive name. For example, you can specify the location name of Receiving Dock to the Ship To business purpose of 100 Main Street.

kit

A BOM Standard Item that has other required BOM Standard Items included in it.

LOV

A list of values in a text field, from which the user must choose.

MACD

An acronym for the following actions: move, add, change, and disconnect. The Oracle Telecommunications Service Ordering solution enables these actions for a customer's telecommunications services.

multiple instantiation

The ability to configure a child Model or Component of a top-level Model multiple times during the same configuration session. The child Model appears in the bill of materials only once, but you can individually configure it as many times as needed while running Oracle Configurator.

network Model

Refer to the Container Model definition.

notifications (notification id)

Notifications are instances of messages which some role receives. The message includes a row that shows status flags to record the state of the notification, date fields for when the notification was sent, due, and responded to. A new row appears in the Notifications table each time a role receives a message. The row persists even after the notification has been responded to, until a purge operation moves to close notifications to an archive.

one-time charge

A charge that a customer pays only one time. Examples include installation fees, activation fees, and change fees.

Oracle Configuration Interface Object (CIO)

A server in the runtime application that creates and manages the interface between the client and the underlying representation of Oracle Configurator Model structure and rules in the generated logic.

The CIO is the API that supports creating and navigating the Model, querying and modifying selection states, and saving and restoring configurations.

partial configuration

The ability to modify part of an existing configuration without launching the entire configuration in Oracle Configurator.

pick-to-order (PTO)

A configure-to-order environment where the options and included items in a Model appear on pick slips and order pickers gather the options when they ship the order. Alternative to manufacturing the parent item on a work order and shipping it. Pick-to-order is also an item attribute that you can apply to standard, Model, and Option Class items.

pick-to-order (PTO) item

A previously defined configuration order that pickers gather as separately finished included items just before they ship the order.

pick-to-order (PTO) Model

An item with an associated bill of material with optional and included items. At order entry, the configurator is used to choose the optional items to include for the order. The order picker gets a detailed list of the chosen options and included items to gather as separately finished items just before order shipment.

provisioning or provisionable

When Oracle Order Management passes the sales order information on to Oracle

Service Fulfillment Manager for provisioning. Provisioning involves capturing an order request, validating the order, analyzing the order, fulfilling the order, completing the order, and if necessary, managing order fallout. When you provision the order, Oracle Install Base receives the reconfigured item information and the sales order is complete.

quote

A collection of items with pricing that a sales representative with a Sales Representative role creates on behalf of a customer in Oracle Quoting.

quote status

The quote status indicates the stage of preparation that a quote is in. Possible quote statuses include drafted, bid, accepted, entered, ordered, order problem, order reviewed, lost, and inactive.

reconfigure

Make changes to a configured Model that a customer has already purchased.

recurring charge

A charge that the customer pays periodically (such as per month, quarter, or year) like a subscription fee.

RMA

Return Material Authorization

SFM

Oracle Service Fulfillment Manager (SFM) provisions the telecommunication services that the customer ordered. See also: provisioning or provisionable.

standard item

An item defined in Oracle Inventory that does not have any components defined in the bill of material (BOM).

tangible item

Any physical BOM Standard Item or Kit that is shippable, inventory transactable and serializable. ATO standard items cannot be tangible items.

TSO

An acronym for the Oracle Telecommunications Service Ordering solution.

tangible item

An item that is Transactable and Shippable. (In the Item definition, these are flagged as Inventory transactable and Shippable).

update

A change to an installed configuration of a Container Model, made in the context of the Telecommunications Services Ordering solution. This document replaces the term with reconfigure. See also: upgrade.

upgrade

In the parlance of the telecommunications service industry, the reconfiguration of a service by moving, adding, changing, or disconnecting items. This document replaces the term with reconfigure. See also: update.

WIP (Work in Progress)

An item in various phases of production in a manufacturing plant. This includes raw material awaiting processing up to final assemblies ready to be received into inventory.

Workflow

This determines the header flow for an order transaction type or line flows possible for a line transaction type. There can be only one header flow associated with an order transaction type but a line transaction type can be coupled with different order types and item types and there can be different flow couplings for the permitted transaction type and item type combinations.

Index

A

access
  discontinued items, 11-34
  instances, 11-34
access level, change Workflow, 15-3
action
  Activate Instance, 10-20
  and Line Types, 3-8
  column, 3-9
  failed, 17-3, 17-4, 17-5
  fulfillment, 11-14
  identify based on, 3-5
  map work item to item and, 15-7
  pricing dependency on Line Type or, 10-21
Action
  See Line Type
actions architecture, 12-2
Activate buttons, adding, 10-20
Activate Instance action, 10-20
Activate Instance UI control, button or picture, 10-20
activation sequence of lines, about, 15-10
active components, 3-5
Active Models
  error when generating, 6-4
  regenerate, 17-2
additional Line Type
  Quoting setup, 5-8
Advanced Pricing
  charge periodicity attribute, 9-2
  display of recurring charges, 9-3
  pricing attributes, 9-3
  setup, 9-1
  TSO functionality, 3-19
API, Oracle Configuration Interface Object, 11-29
APIs
  COPY_CONFIGURATION, 10-30
  COPY_CONFIGURATION_AUTO, 10-36
  VALIDATE, 10-36
ASO: Allow Quantity Updates for Component Item, 14-2
ASO: Allow Quantity Updates for Top Level Model Item, 14-2
ASO: Default IB Search Results, 14-2
ASO: Default Install Base Relationship, 14-3
ASO: Default Ordered Qty in OC UI, Forms Quoting, 16-19
ASO: Default Order Quantity in OC UI, 14-3
ASO: Default Order Type, 14-3
ASO: Default Quote Status, 14-3
ASO: Display Installed Base Attributes, 14-3
ASO: Max Number of IB Search Results, 14-3
ASO: Price List Override, 16-19
ASO: Use Network Container, 14-4
  Forms Quoting, 16-22
ATO BOM Models, not included in TSO solution, 3-4
Attribute Code, 10-23
attributes, 10-23
  code, 10-23
  configuration, using with IB, 11-26
  extended, 8-2
  extended attribute pools, 8-3
  item category level, 8-3

mandatory, 15-10
map extended, to SFM items, 8-4
synchronization after fulfillment, 11-36
verify coordination of, 8-5
what is an extended, 15-11
attribute values
changing user, A-7
compared with IB tables, 10-24
item instances, A-2
price based on, 9-4
saved in configuration, 11-28
used to select, 9-7

B

baselines
configuration, identify and maintain, 3-6
configuration, status, 2-4
current state of configuration, 3-5
identify current configuration, 3-6
methods to identify changes against, 11-30
show changes since, 2-4
upgrades and configuration, 3-4
batch validation, 3-8
batch validation parameter, 10-29
Bills of Material
setup, 7-1
Bills of Material (BOM)
activation sequence of lines, 15-10
Model, component tracking, specify, 6-5
Model node, 10-11
Option Class, component tracking, specify, 6-5
Standard Item, component tracking, specify, 6-5
booked orders, about copying, A-2
business flows, 2-1
buttons
Activate Instance UI control, 10-20
adding Activate, 10-20
create Connection List, 10-20
Test Model, 11-12, 11-25, 11-28

C

change
configuration, 3-8
detecting type of, 11-13
identify configuration, 11-30
multiple instantiable child Models, A-5
required revalidation to substantial, A-5
since baseline configuration, 2-4
user attribute in SFM, A-7
validation of, 2-4
workflow access level, 15-3
charge periodicity
defined, 3-1
pricing attribute, 9-2
Charge periodicity
defined, 5-3
Charge Periodicity Code column, 5-3
child Model
about deleting trackable, 17-1
import process creates structure for, 10-6
trackable, structure of, 10-6
tracking multiple instantiable, A-5
valid configuration rule combinations, 10-16
child of Component, 10-3
child of Model, 10-3
code
Attribute, 10-23
customize, 11-17
Location and Location Type, 3-7
Location Type, 10-27, 11-6, 11-6
message, 17-4
units of measurement, 10-27
column
determine display of Line Type, 10-24
component instances
about, 11-29
in Installed Base, track, 11-29
multiple, 11-29
component items
specify, 6-5
tracking, 6-5
components
active, 3-5
passive, 3-5
concurrent programs
Disable/Enable Refresh of a Configuration Model, 10-5
Import Configuration Models, 10-19
Populate Configuration Models, 10-5, 10-7
Refresh a Configuration Model, 10-5, 10-6
Refresh all Imported Configuration Models, 10-5, 10-6

under Populate and Refresh Configuration Models, 10-5
XDP Resubmit Failed Message, 17-4
configurable items
network link, 6-6
setup, 6-2
configuration
as item instances, 3-6
attribute, CZ_CONFIG_EXT_ATTRIBUTES table, 10-24
attributes with Installed Base, 11-26
automatic behavior of, 11-35
baseline, current state of, 3-5
changes to, identify, 11-30
CSI_T_TRANSACTION_LINES, update, 10-26
CSI_T_TXN_LINE_DETAILS, update, 10-26
delta computation, 3-5
identify current baseline, 3-6
invalid rule combinations, 10-17
make changes to, 3-8
Model, create, 10-19
multiple component instances, 11-29
pricing, 2-8
rules, define, 10-19
show changes since baseline, 2-4
store Container Model, 3-6
update, 11-29
valid rule combinations, 10-16
verify invalid items, 17-2
configuration changes
CIO methods, 11-30
Configuration Model Type, 10-6
Configurator
configuration Model, create, 10-19
Container Model settings and structure, 10-4
Container Model structure, 10-6
customize, 10-1, 11-1
dependencies, 10-1
functionality, 3-4
import Container Models into, 10-5
locking instances, 2-8
map Line Type, 5-2
setup, 10-1, 11-1
validate changed configuration, 2-4
Configurator Extension
IBAttribute, about, 10-20
IBAttribute, effects of, 11-28
IBAttribute, modify, 11-28
IBAttribute, setup, 11-26
Line Type, about, 10-20
Line Type, modify, 11-26
Line Type, setup, 11-13
Line Type, specify, 11-13
Line Types, effects of, 11-25
Location, 10-20
configure, new items, 2-1
connected-to relationships, 3-6
connected-to relationships of configured instance, view, 3-6
Connection List buttons, 10-20
connection rule, adding, deleting, or changing, A-5
Connector
define rules for, 10-10
define rules using, 10-16
in Container Model, 10-9
node, delete, A-5
reasons not to create, 10-10
requirements to create, 10-10
Connector node, 10-3
Contact Center
actions architecture, 12-2
setup, 12-1
using, 16-1
container BOM Model
import, 10-6
See also Container Model.
container BOM Model, 6-5
tracking for components, specify, 6-5
Container Model
about rules when revalidating, 17-1
and kit items, 3-9
Connector requirements, 10-9
create, 6-4
define, 7-1
define as PTO BOM Model requirement, 10-4
description, 10-3
error when trackable, 6-4
flag as, 6-2
import, 10-19
import into Configurator, 10-5
installed configurations, revalidate, 17-1
node, 10-17
node from non-trackable child, 10-18

PTO (pick-to-order), 7-1
publish, 10-20, 10-20
refresh error, 10-6
refresh warning, 10-6
requirements, verify, 6-5
requirements not to define, 10-4
root node, 10-10
settings and structure, 10-4
specify, 6-4
specify item as, 6-4
store, 3-6
structure, 10-6
structure requirements, 10-6
tracking for component item, specify, 6-5
valid structure of, verify, 10-19
convert
cannot create order from quote, A-2
coordinate, attributes and extended attributes, 8-5
COPY_CONFIGURATION_AUTO (API), 10-36
COPY_CONFIGURATION (API), 10-30
create
configuration Model, 10-19
Container Model, 6-4
workflow process, 15-5
credit check rule setup, 5-6
CSI_T_TRANSACTION_LINES, mapping to CZ for configuration updates, 10-26
CSI_T_TXN_LINE_DETAILS, mapping to CZ for configuration updates, 10-26
CSI: Configurator Enabled, 8-2
customer, updating, when selecting item instance, A-6
customize
code, 11-17
solutions, 11-3
TSO, 11-3
user interface, 10-20
workflow, 15-3
CZ_CONFIG_EXT_ATTRIBUTES, Table in CZ schema, 10-24
CZ_CONFIG_ITEMS, mapping to the Install Base schema, 10-26
CZ_DB_SETTINGS (database table)
SETTING_ID
GenerateUpdatedOnly, 10-18
CZ: Auto-Expire Discontinued IB Trackable Items, profile option, 10-21
CZ: Configurator Install Base, profile option, 10-21
CZ: Enable ATP, profile option, 10-22
CZ: Enable List Prices, profile option, 10-22
CZ: Enable Selling Prices, profile option, 10-22
CZ: Include Unchanged Install Base Items, profile option, 10-21
CZ: Only Create CZ Config Items for Selected Nodes, 3-9
CZ: Only Create CZ Config Items for Selected Nodes, profile option, 10-21
CZ: Publication Usage, 13-5
CZ: Report All Baseline Conflicts, profile option, 10-21
CZ: Suppress Baseline Errors, profile option, 10-21

D

data, out of synchronization, 17-1
default
Line Type, 14-6
defaulting rules
payment term, 5-5
payment type, 5-6
define
configuration Model, 10-19
Container Model, 7-1
Container Model as non-trackable, 10-3
Container Model as PTO BOM Model requirement, 10-4
Container Model do not, 10-4
items as link items, 3-6
Line Types, 10-20, 11-14, 17-2
profile options, 10-21
rules for Connectors, 10-10
rules using Connectors, 10-16
service items, 6-2
work item, 15-6
delete
Connector node, A-5
trackable child Model, impact of, 17-1
trackable multiple instantiable child Model, A-5
Disable/Enable Refresh of a Configuration Model concurrent program, 10-5

discontinued items
access, 11-34
CIO methods, 11-34
definition, 11-33
discounts
on one-time charges, 3-3
on recurring charges, 3-2

E

Enable Recurring Charges system parameter, 5-3
Error Item Type, map, 15-6
Error Process, map, 15-6
errors
attempt to copy, 10-35
Container Model marked as trackable, 6-4
copy failure, 10-31
generate logic in Configurator Developer, 10-6
Install Base update failure, 17-4
manage failure notification, 17-5
manage notification, 17-3
message, failure to accept order, 15-10
no configuration, 10-31
notification related, 17-3
Provisioning Failed, 15-10
retry failed action, 17-4
retry failed outgoing messages, 17-4
sales order notification, 17-3
troubleshooting copy failure, 10-35
update configuration when re-validating, 17-3
view notification, 15-3
when generating logic, 10-6, 10-7, 10-10, 10-10, 10-11, 10-17, 10-18, 10-18
when importing Container Model, 6-4
with child BOM Models, 10-6
with Option Feature maximum, 10-13
event manager queue, 8-4
events
for Configurator Extensions, 11-35
onInstanceLoad, 11-36
postInstanceLoad, 11-36
related to TSO, 11-35
extended attributes, 8-2
CIO methods, 11-37
item category levels, 8-3
map to SFM items, 8-4
verify coordination of, 8-5
what are, 15-11

F

failed action, 17-3, 17-4, 17-5
Financials integration, 2-3, 3-8
find
item instances, A-6
order, 17-3
flag, item as Container Model, 6-4
Fulfilled status, change to, 2-8
fulfillment
action, 11-14
actions apply, 11-13
and Line Types, 3-5
based on actions, 3-5
building workflows, A-5
change user attribute values, A-7
creating attribute data, 11-36
date, 5-9
date required, 5-9
Line Type activities, 10-20
map transient attributes, 10-22
node, 5-9
of provisionable items, 5-9
one-time, 10-11
persistence of user-defined values, 3-6
quantity, 5-9
restoring attribute data after, 11-37
sequence of, A-7
set Location and Location Type Code, 3-7
synchronization of attributes, 11-36
synchronization of instance names, 11-37, 11-37
update of Installed Base, 5-9
validating status, 10-29
validating status of, 10-36
workflow item, 15-2

G

GenerateUpdatedOnly
CZ_DB_SETTINGS, 10-18
generate user interface, 10-19

H

header folders modification, 5-7

hosting application, defined, 3-7
hosting application, specify, 10-20

I

IBAttribute
  Configurator Extension, 10-20
  effects of, 11-28
  modify, 11-28
  setup, 11-26
IBE: Autoquery Install Base for B2C Users, 13-4
IBE: Default Payment Term, 13-5
IBE: Display Option Classes in the Shopping Cart and Order Tracker, 13-3
IBE: Enable Install Base View, 13-3
IBE: Enable Pay Now, 13-3
IBE: Enable Recurring Charges, 13-3
IBE: Item Validation Organization, 13-5
IBE: View Only Web Published Items in Install Base, 13-4
implementation flow, 4-5
implementation overview, 4-1
import
  container BOM Model, 10-6
  Container Model, 10-19
  create child Model structure, 10-6
  data with Populate and Refresh Configuration Models, 10-5
  Populate Configuration Models, 10-5
  refresh BOM with sub-models, 10-6
Import Configuration Models concurrent program, 10-19
indirection, for specifying Line Type data, 11-23
initialization parameters
  about, 10-28
  multiple times, specify, 10-29
Install Base
  retry updates to, 17-4
  update failure, 17-4
Install Base schema
  CSI_T_TRANSACTION_LINES, 10-26
  CSI_T_TXN_LINE_DETAILS, 10-26
Installed Base
  and iStore, 3-9
  and sales agent applications, 3-9
  configuration attributes with, 11-26
  extended attributes, 8-2
  functionality, 3-6
  integration with Configurator, 3-4
  item, track in, 6-5
  search, 3-9
  setup, 8-1
  Tracking check box, 6-5
installed configuration, reconfigure, 10-20
Installment Options system parameter, 5-5
Instance Class, set to Link, 6-5
instance names
  synchronization after fulfillment, 11-37, 11-37
instances
  access to, 11-34
  CIO methods, 11-34
  component, 11-29
  locked, 10-15
  minimum and maximum, PTO (pick-to-order), 10-7
  multiples of configurable components, 11-29
  trackable, 10-6
instantiability maximum, 10-7, 10-7
instantiable child Model
  delete trackable multiple, A-5
  track multiple, A-5
intangible item
  defined, 3-9
Inventory
  Container Model, 6-4
  Container Models, 6-4
  define items as link items, 3-6
  setup, 6-1
  tracking for component item, specify, 6-5
iStore
  setup, 13-1
  TSO functionality flows, 3-11
  TSO item display, 16-3
item category extended attribute levels, 8-3
Item Instance Query window, 3-7
item instances
  attribute values, A-2
  find, A-6
  querying, 3-7
  reconfigured, copying booked order, A-2
  reconfigured, copying order lines, A-2
  removing order lines, A-2
  revision of, A-2
  saved configuration appears as, 3-6

selected for reconfiguration, A-5
updating a customer, A-6
view or reconfigure configured, 3-6
items
configure new, 2-1
intangible, 3-9
setup for iStore, 13-5
setup for Payment Due with Order amount, 5-6
tangible, 3-8

K

kit
as part of Container, 3-9

L

Line Payments form setup, 5-7
Line Type
and actions, 3-8
and changes, 3-5
column, determine display of, 10-24
Configurator Extension, 10-20, 11-13
customized, 5-8
default, 14-6
define, 10-20, 11-14, 17-2
effects of, 11-25
in Order Management, 2-5
mapping, 5-2
missing nodes, 11-18
modify, 11-26
No Action, 17-2
pricing dependency on action or, 10-21
provisionable item includes mapped, 15-11
Reprice, 17-2
set up, 11-13
type of change, 2-4
unchanged nodes, 11-19
LINETYPE_MISSING_NAME (Property name), 11-19
LINETYPE_MISSING (Property name), 11-19
LINETYPE_n_ID (Property name), 11-15
LINETYPE_n_NAME (Property name), 11-15
LINETYPE_n_TEST (Property name), 11-15
LINETYPE_UNCHANGED_NAME (Property name), 11-20
LINETYPE_UNCHANGED (Property name), 11-20
Line Type Configurator Extension, 11-25
link, specify service item as, 6-2, 6-6
link flag, schema change, 10-28
link items
CIO methods, 11-35
location
background, 11-6
Configurator Extension, 10-20
Location and Location Type Code, 3-7
Location Type Code, 10-27, 11-6, 11-6
logic
error when generating, 10-6, 10-7, 10-10, 10-10, 10-17, 10-18, 10-18
generating, 10-19
no error when generating, 10-11
lookup codes
SFM Lookup, 15-6

M

MAC
See message authentication code
manage, failure notifications, 17-5
mandatory attribute, 15-10
mandatory dependencies, 4-2
map
item and Line Type combinations to work item, 15-11
provisionable item to Line Type, 15-11
menu notification, add function for, 15-3
message authentication code (MAC), 11-12
message code, 17-4
messages
failure to send, 17-5
monitor and verify sending of, 17-5
retry failed outgoing, 17-4
MO: Operating Unit, 13-5
model, container, 10-3
model data, out of synchronization, 17-1
Model structure, 10-3
modify
IBAttribute, 11-28
Line Type, 11-26
My Products screen, 3-9

N

NAME (database column), 11-14
Negotiation phase, converting an order in, A-2
network configuration models
  storing, 3-6
network links
  service item, 6-6
  service item as, specify, 6-2
networks
  partial reconfiguration and validation, 3-4
  storing configuration model, 3-6
No Action Line Type, 17-2
nodes
  added to session, 11-30
  BOM Model, 10-11
  changed configuration attributes, 11-31
  changed descendant, 11-31
  changed location or location type code, 11-31
  Container Model, 10-17
  Container Model's root, 10-10
  deleted from session, 11-30
  from non-trackable child, 10-18
  in Configurator, creation of model, 10-6
  in parent model, reference, 10-6
  value change, 11-30
notifications
  add function for menu, 15-3
  errors, manage, 17-3
  manage failure, 17-5
  retry failed action, 17-4
  retry failed outgoing messages, 17-4
  retry Install Base updates, 17-4
  sales order error, 17-3
  view error, 15-3
  with Open status, 17-4

O

OM: Sales Order Form: Cascade Header Changes to Line, 5-8
OM: UOM Class for Charge Periodicity, 5-7
one-time charges
  and Advanced Pricing, 3-2
  discounts on, 3-3
  display of totals, 3-3
  modeling options, 3-2
onInstanceLoad (event), 11-36
ONT.OE_TRANSACTION_TYPES_TL (database table), 11-14
Open status, 17-4
optional integrations, 4-5
Oracle Applications Framework, 11-12
Oracle Configuration Interface Object, about, 11-29
Oracle Configurator. See Configurator., 10-1, 11-1
Order Flow-Through area, 17-3
order lines
  about copying, A-2
  removing, A-2
  status of provisionable, 15-10
Order Management, 16-23
  assign workflows to transaction types, 5-7
  cascade header changes to line, 5-8
  create header and line workflows, 5-8, 5-9
  dependencies, 5-1
  map Line Type, 5-2
  publication available to, 10-20
  replace fulfill node, 5-9
  set profile options, 5-7
  setup, 5-1
  Workflow Header Process, SFM enabled, 15-2
  Workflow Line Process, 15-3
orders
  booked, about copying, A-2
  cannot convert quotes to, A-2
  convert in Negotiation phase, A-2
  find, 17-3
  provisioning, 2-8
  search, 17-3
  sequence of fulfilling, A-7
  stacked, 10-15
Outbound Message Queue, 17-5

P

parameters, batch validation, 10-29
passive components, 3-5
Pay Later
  functionality, 3-3
Payment Due with Order
  setup, 5-4
Payment Due with Order amount
  iStore setup, 13-2
  overview, 3-2
  payment term setup, 5-5

setup in Quoting, 14-5
taxes setup, 5-7
Payments form setup, 5-7
payment term
for Payment Due with Order amount, 5-5
set up defaulting rule, 5-5
payment type: set up defaulting rule, 5-6
pending orders
making changes to, A-1
periodicity
set profile option, 5-7
setup, 5-3
pick-to-order. See PTO., 6-4
PL/SQL
procedures
COPY_CONFIGURATION, 10-30
COPY_CONFIGURATION_AUTO, 10-36
VALIDATE, 10-36
Populate and Refresh Configuration Models
concurrent programs, 10-5
import data with, 10-5
Populate Configuration Models concurrent program, 10-5, 10-7
postInstanceLoad (event), 11-36
prevent
configuration model refreshing, 10-5
price lists
determine which to use, 9-3
source of pricing attributes, 9-3
pricing
attribute-based setup, 9-4
attribute linking and mapping, 9-4
attributes and sourcing rules, 9-3
based on attribute value, 9-4
dependency on Line Type or action, 10-21
disable, 10-21
setup in iStore, 13-7
pricing attributes
source data for, 9-3
pricing attributes, through price lists, 9-3
profile option
CZ: Auto-Expire Discontinued IB Trackable Items, 10-21
CZ: Configurator Install Base, 10-21
CZ: Enable ATP, 10-22
CZ: Enable List Prices, 10-22
CZ: Enable Selling Prices, 10-22
CZ: Include Unchanged Install Base Items, 10-21
CZ: Only Create CZ Config Items for Selected Nodes, 10-21
CZ: Report All Baseline Conflicts, 10-21
CZ: Suppress Baseline Errors, 10-21
define, 10-21
progress, an order, 15-10
provisionable
item, includes mapped Line Type, 15-11
item, set service item as, 6-6
item, specify service item as, 6-2
order line, status of, 15-10
Provisionable, check box, 6-5
provisioning
sales order, 2-8
status change after, 2-8
provisioning action, 2-2, 2-5
Provisioning Failed
status, 15-10
status on sales order form, 17-4
Provisioning Successful status
change to, 2-8
PTO (pick-to-order)
BOM Model, 6-4
Container Model, 7-1
minimum and maximum instances, 10-7
publication, available to Order Management, 10-20
publish, Container Model, 10-20, 10-20

Q

Quoting
dependencies, 14-1
design tips, 14-6
functionality, 3-7
Lines tab stup, 14-6
modifying the UI, 14-5
remove unchanged components, 16-15, 16-22
setup, 14-1
TSO functionality flow, 3-9
using, 16-13, 16-18

R

reconfiguration

definition, 2-5
reconfigure
and removing unchanged components, 16-15
Container Model after booking, 5-10
installed configurations, 10-20
installed instances, 2-3, 3-8
item instances, 3-6
partial network, 3-4
reconfigured item instances
copying booked orders, A-2
copying order lines, A-2
recurring charges, 3-3
and Advanced Pricing, 3-1
charge periodicity pricing attribute, 9-2
discounts on, 3-2
display in Advanced Pricing reports, 9-3
in iStore, 13-2
modeling options, 3-2
setup in Order Management, 5-3
setup in Quoting, 14-4
Reference node, 10-3
refresh
instantiable models, 10-6
models with references, 10-6
prevent configuration model, 10-5
Refresh a Configuration Model concurrent program, 10-6
Refresh all Imported Configuration Models concurrent program, 10-5, 10-6
Refresh a Single Configuration Model concurrent program, 10-5
Reprice Line Type, 17-2
return status, 10-31
revalidate, installed configurations, 17-1
revision of item instance, A-2
rule
containing participants, 17-1
invalid configuration combinations, 10-17
valid configuration combinations, 10-16
when re-validating Container Model, 17-1

S

sales agent applications
Installed Base integration, 3-9
sales agent applications, defined, 3-7
sales orders
monitor for error notifications, 17-3
search
for order, 17-3
Service Fulfillment Manager
change to user attribute, A-7
design tips, 15-10
fails to accept order, 15-10
functionality, 3-19
key functionality, 3-19
map extended attributes to, 8-4
map work item, 15-7
provisioning in, 2-8
setup, 15-1
set up event manager queue, 8-4
Workflow Header Process, 15-2
workflow process, create, 15-5
work item, define, 15-6
service items
network link, 6-2
provisionable items, 6-2
set provisionable, 6-6
services
as Container Model, flag, 6-2
setup
Bills of Material, 7-1
configurable items, 6-2
Configurator, 10-1, 11-1
Contact Center, 12-1
Installed Base, 8-1
Inventory, 6-1
iStore, 13-1
Order Management, 5-1
Quoting, 14-1
Service Fulfillment Manager, 15-1
SFM. See Service Fulfillment Manager., 3-19
SFM Lookup Codes, 15-6
short names, of Oracle Applications, 10-20
solutions, customize, 11-3
status
after provisioning, 2-8
Fulfilled, change to, 2-8
monitor area, 17-5
of provisionable order line, 15-10
Open, 17-4
Provisioning Failed, 15-10
provisioning failure on sales order form, 17-4
record, 10-3

return, 10-31
validating fulfillment, 10-29, 10-36
Status Monitor area, 17-5
substantial change, required revalidation, A-5
summary window, 2-4
synchronization, data, out of, 17-1
system parameters for Order Management, 5-5

T

tangible item, 3-4
CIO methods, 11-37
Container Model, 10-5
defined, 3-8, 10-14
enabling, 6-5
importing, 10-6
reconfiguration of , 3-9
restrictions, 10-14
schema change, 10-28
taxes
setup for Payment Due with Order amount, 5-7
Telecommunications Billing Integrator
integration, 2-8
Telecommunications Billing Integrator
functionality, 2-3, 3-8
Test Model button, 11-12, 11-25, 11-28
trackable
delete trackable child Model, 17-1
instance, 10-6
make child Model non-trackable, 17-1
TRANSACTION_TYPE_ID (database column), 11-14
transaction type, about, 10-20
transient flags
designate items in configuration model, 10-12
schema change, 10-28
setting, 10-19
transient item
and one-time charges, 3-2
defined, 3-2
transient items
definition, 10-11
TSO functionality
applications involved, 1-2
business processes, 2-1
core functionality by application, 4-2
iStore flows, 3-11
overview, 1-1, 3-1
Quoting flow, 3-9

U

units of measurement (UOM)
code, 10-27
update, 11-29
See reconfiguration
customer when selecting item instance, A-6
installed configurations, 11-29
upgrade
See reconfiguration
usages in iStore, 13-7
user interface
customize, 10-20
generate, 10-19
user setup for iStore, 13-7
user tasks, 16-1

V

VALIDATE (API), 10-36
validating fulfillment status, 10-36
validation
batch validation, 3-8
validation, partial network, 3-4
verify
Container Model requirements, 6-5
coordination of attributes, 8-5
invalid items within configuration, 17-2
Line Type, 17-2
sending of messages, 17-5
valid Container Model structure, 10-19
view
error notifications in HTML, 15-3
messages, 17-5
Service Fulfillment lines, 17-3

W

Workflow Header Process, create SFM Enabled OM, 15-2
workflow line process, create, 15-3
workflows
change access level, 15-3
create header and line, 5-9

customize, 15-3
process, create, 15-5
work items
define, 15-6
item and Line Type combinations map to, 15-11
map to action combination, 15-7
parameter, correct, 17-5

X

XDP Resubmit Failed Message concurrent program, 17-4