Oracle® Contracts Core

Implementation Guide

Release 11*i*

Part No. B10678-01

July 2003

ORACLE®

Oracle Contracts Core Implementation Guide, Release 11*i*

Part No. B10678-01

Contents

Send Us Your Comments

Oracle Contracts Core Implementation Guide, Release 11*i*

Part No. B10678-01

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, please indicate the document title and part number, and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- lectronic mail: Contracts@oracle.com
- FAX: 650-633-3782 Attn: Oracle Contracts Core
- Postal service
  Oracle Corporation
  Oracle Contracts Core Documentation Manger
  500 Oracle Parkway
  Redwood Shores, CA 94065
  U.S.A.

If you would like a reply, please give your name, address, telephone number, and (optionally) electronic mail address.

If you have problems with the software, please contact your local Oracle Support Services.

Preface

Welcome to the **Oracle Contracts Core, Release 11*i***. This Implementation Guide provides information and instructions to help you work effectively with Oracle Contracts Core.

This preface explains how Implementation Guide is organized and introduces other sources of information that can help you.

Intended Audience

This guide is aimed at the following users:

- Implementation Team Members
- Technical Service Representatives (TSR)
- Customer Service Representatives (CSR)
- System Administrators (SA), Database Administrators (DBA), and others with similar responsibility.

This guide assumes you have the following pre-requisites:

1. Understanding of the company business processes.
2. Knowledge of products and services as defined by your marketing policies.
3. Basic understanding of Oracle and Developer/2000.
4. Background in SQL, PL/SQL, SQL* Plus programming.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. 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. Standards will continue to evolve over time, and Oracle Corporation 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 additional information, visit the Oracle Accessibility Program Web site at `http://www.oracle.com/accessibility/`.

Accessibility of Code Examples in Documentation JAWS, a Windows screen reader, 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, JAWS 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 Corporation does not own or control. Oracle Corporation neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Structure

This manual contains the following chapters:

Chapter 1 Introduction

This chapter presents an overview of Contracts Core, the technology stack, architecture, and seed data protection.

Chapter 2 Implementation Prerequisites

This chapter presents mandatory dependencies to Contracts Core and prerequisite setups for other Oracle modules.

Chapter 3 Implementing Contracts Core

This chapter provides the procedures for the setup and configuration tasks required to implement Contracts Core successfully.

Chapter 4 Customizing Contracts Core

This chapter contains procedures for customizing events, workflows, and registering a new source as a JTF object.

Chapter 5 Integrating Contracts Core with Oracle Pricing

This chapter presents the procedures for upgrade and setup for integration to Oracle Pricing.

Chapter 6 Integrating Contracts Core with Oracle Configurator

This chapter covers integration and related setups to Oracle Configurator.

Appendix A Contracts Core Profile Options

This appendix contains a table displaying the Contracts Core profile options.

Appendix B Setting up Lookup Codes

This appendix contains a table displaying the Contracts Core lookup codes.

Other 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 Contracts Core.

If this guide refers you to other Oracle Applications documentation, use only the Release 11*i* versions of those guides.

Online Documentation

All Oracle Applications documentation is available online (HTML or PDF). Online help patches are available on Oracle*MetaLink*.

Related Documentation

Oracle Contracts Core shares business and setup information with other Oracle Applications products. Therefore, you may want to refer to other product documentation when you set up and use Oracle Contracts Core.

You can read the documents online by choosing Library from the expandable menu on your HTML help window, by reading from the Oracle Applications Document Library CD included in your media pack, or by using a Web browser with a URL that your system administrator provides.

If you require printed guides, you can purchase them from the Oracle Store at http://oraclestore.oracle.com.

Documents 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) available with this release of Oracle Contracts Core (and any other Oracle Applications products). This guide also includes information on setting user profiles, as well as running and reviewing reports and concurrent processes.

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

Documents Related to This Product

Oracle Contracts Core User Guide

This user guide explains the concepts and procedures for using Oracle Contracts Core.

Oracle Sales Contracts Concepts and Procedures

Use this manual to learn more about using the Sales Contractst functionality in Oracle Contracts Core.

Installation and System Administration

Oracle Applications Concepts

This guide provides an introduction to the concepts, features, technology stack, architecture, and terminology for Oracle Applications Release 11*i*. It provides a useful first book to read before an installation of Oracle Applications. This guide also introduces the concepts behind Applications-wide features such as Business Intelligence (BIS), languages and character sets, and Self-Service Web Applications.

Installing Oracle Applications

This guide provides instructions for managing the installation of Oracle Applications products. In Release 11*i*, much of the installation process is handled using Oracle Rapid Install, which minimizes the time to install Oracle Applications, the Oracle8 technology stack, and the Oracle8*i* Server technology stack by automating many of the required steps. This guide contains instructions for using Oracle Rapid Install and lists the tasks you need to perform to finish your installation. You should use this guide in conjunction with individual product user's guides and implementation guides.

Oracle Applications Supplemental CRM Installation Steps

This guide contains specific steps needed to complete installation of a few of the CRM products. The steps should be done immediately following the tasks given in the Installing Oracle Applications guide.

Upgrading Oracle Applications

Refer to this guide if you are upgrading your Oracle Applications Release 10.7 or Release 11.0 products to Release 11*i*. This guide describes the upgrade process and lists database and product-specific upgrade tasks. You must be either at Release 10.7 (NCA, SmartClient, or character mode) or Release 11.0, to upgrade to Release 11*i*. You cannot upgrade to Release 11*i* directly from releases prior to 10.7.

Maintaining Oracle Applications

Use this guide to help you run the various 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 Applications System Administrator's Guide

This guide provides planning and reference information for the Oracle Applications System Administrator. It contains information on how to define security, customize menus and online help, and manage concurrent processing.

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 6*i* forms so that they integrate with Oracle Applications.

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 how to apply this UI to the design of an application built by using Oracle Forms.

Other Implementation Documentation

Oracle Workflow Guide

This guide explains how to define new workflow business processes as well as customize existing Oracle Applications-embedded workflow processes.You also use this guide to complete the setup steps necessary for any Oracle Applications product that includes workflow-enabled processes.

Oracle Applications Flexfields Guide

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

Oracle eTechnical Reference Manuals

Each eTechnical 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, integrate Oracle Applications data with non-Oracle applications, and write custom reports for Oracle Applications products. Oracle eTRM is available onOracle*MetaLink*

Oracle Manufacturing APIs and Open Interfaces Manual

This manual contains up-to-date information about integrating with other Oracle Manufacturing applications and with your other systems. This documentation includes APIs and open interfaces found in Oracle Manufacturing.

Oracle Order Management Suite APIs and Open Interfaces Manual

This manual contains up-to-date information about integrating with other Oracle Manufacturing applications and with your other systems. This documentation includes APIs and open interfaces found in Oracle Order Management Suite.

Oracle Applications Message Reference Manual

This manual describes Oracle Applications messages. This manual is available in HTML format on the documentation CD-ROM for Release 11i.

Oracle CRM Application Foundation Implementation Guide

Many CRM products use components from CRM Application Foundation. Use this guide to correctly implement CRM Application Foundation.

Training and Support

Training

Oracle offers training courses to help you and your staff master Oracle Contracts Core and reach full productivity quickly. You have a choice of educational environments. You can attend courses offered by Oracle University at any one 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's 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 Oracle Contracts Core 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 Oracle8*i* server, and your hardware and software environment.

Oracle*MetaLink*

Oracl*eMetaLink* is your self-service support connection with web, telephone menu, and e-mail alternatives. Oracle supplies these technologies for your convenience, available 24 hours a day, 7 days a week. With Oracl*eMetaLin*k, you can obtain information and advice from technical libraries and forums, download patches, download the latest documentation, look at bug details, and create or update TARs. To use MetaLink, register at (http://metalink.oracle.com).

Alerts: You should check Oracl*eMetaLink* alerts before you begin to install or upgrade any of your Oracle Applications. Navigate to the Alerts page as follows: Technical Libraries/ERP Applications/Applications Installation and Upgrade/Alerts.

Self-Service Toolkit: You may also find information by navigating to the Self-Service Toolkit page as follows: Technical Libraries/ERP Applications/Applications Installation and Upgrade.

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 Oracle Applications 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.

About Oracle

Oracle Corporation develops and markets an integrated line of software products for database management, applications development, decision support, and office automation, as well as Oracle Applications, an integrated suite of more than 160 software modules for financial management, supply chain management, manufacturing, project systems, human resources and customer relationship management.

Oracle products are available for mainframes, minicomputers, personal computers, network computers and personal digital assistants, allowing organizations to integrate different computers, different operating systems, different networks, and even different database management systems, into a single, unified computing and information resource.

Oracle is the world's leading supplier of software for information management, and the world's second largest software company. Oracle offers its database, tools, and applications products, along with related consulting, education, and support services, in over 145 countries around the world.

1

Introduction to Oracle Contracts Core Implementation

This chapter provides an overview to Oracle Contracts Core as it relates to implementation. The topics discussed include:

- Oracle Contracts Suite Overview
- Oracle Contracts Core Overview
- Technology and Architecture

1.1 Oracle Contracts Suite Overview

Oracle Contracts is a suite of applications designed to centralize and provide visibility and management for contracts of all types within a company. Oracle Contracts Suite streamlines enterprise contract management, providing for contracts to be composed, administered, tracked, and renewed systematically and consistently.

The following applications represent the suite:

- Contracts Core
- Sales Contracts
- Service Contracts
- Rights Contracts
- Contracts Intelligence

1.2 Oracle Contracts Core Overview

Oracle Contracts Core is designed for any business that needs to manage sales and procurement contracts across their complete enterprise contract management life cycle. Oracle Contracts Core provides a central repository for collection and dissemination of contract information and is responsible for all common functionality across the contracts suite, such as renewing or extending a contract, requesting a contract change, managing contract deliverables, or handling the contract-approval process.

1.3 Technology and Architecture

Technology and architecture covers the following topics:

- Technology Stack
- Architecture
- Seed Data Protection

1.3.1 Technology Stack

Oracle E-Business Suite is an integrated set of business capabilities that is also an integrated set of technologies. The technology stack consists of:

- Oracle Database and related applications
- Internet application server
- Oracle Workflow
- Oracle security technology
- Oracle XML
- Oracle Discoverer Reports

1.3.2 Architecture

The following components describe the contracts architecture:

Reference:
Oracle Contracts Core API Guide

1.3.2.1 API and View Layer

Two primary types of PL/SQL APIs:

- Entity
- Process

Three levels of Entity APIs:

- Public
- Complex
- Simple

Two levels of Process APIs

- Public
- Process

UI, external systems interface to public API layer. Insert, update, and delete operations are through APIs.Views cover tables. Query access through views.

1.3.2.2 Entity Relationship Model

Class and Category:

- A contract class is a high level determination of the contract type. Classes are seeded. Examples of classes include: Service, Corporate, and Rights
- Contract category is a sub-division within a class. Contract category drives many aspects of the contract, such as what line style hierarchies are allowed, what party and contact roles are allowed, and what rule groups are allowed at either the header or line level. Examples of categories can include, sales agreement, warranty and extended warranty, and network license.

Contract Header:

- Holds common attributes such as contract number, start and end dates, contract category, status, intent and perspective.

Contract Lines:

- "Top Lines" generally specify what the contract is about, such as which services have been purchased or what software is licensed.
- "Sublines" further refine their parent line, e.g. which products are covered by a service, and also show configured items related to top line item.

- Depending on the contract category, both top lines and sublines may have duration and status.
- Lines have "Line Styles". A line style hierarchy shows what types of lines are allowable as child sublines.
- Allowable items, party roles, and rule groups per category are assigned to top line in line style hierarchy.

Parties and Contacts:

- Contract parties are parties or accounts in the Trading Community Architecture (TCA), vendors, or organizations.
- Contract parties play roles in the contract.
- Contacts also play roles and must be associated with a contract party.
- The availability of a party or contact may be constrained by how it is to be used. For example, a billing contact must be correctly defined in AR.

Articles:

- Articles are the text of the contract.
  - Standard articles are kept in a Standard Article Library and are used in many contracts.
  - Custom articles are created in the context of one contract.
- A contract does not need to have any articles.

Rules:

- Rules are the structured data representation of terms and conditions of the contract, intended for use by automated processes.
- Rules are organized into rule groups, such as rules guiding the billing process would be in a billing rule group. Rule groups may be thought of as business processes.
- Rule groups may be associated with the header or a line.

Status and Operation:

- Status and operation controls what operations are allowable for a contract in a certain status. It is variable by contract category.
- Examples of operations are: update online and update via change request.

Data Sources:

- Defines the source for referenced information, by contract category and sometimes by buy/sell.
- Provides flexible mechanism for sourcing referenced information.
- Used for rules, parties, contacts, and lines.

1.3.2.3 Organizations

A contract has an "authoring" operating unit.

- All referenced organization-partitioned data is from the authoring organization.
- Users whose responsibility is in another organization and who may update the contract, do so in the context of the authoring organization.

1.3.2.4 Security

Security is by either of two mechanisms:

- An access control list based on JTF resources and resource groups, specifies who may view or update the contract.
- By responsibility and category: You can specify that a responsibility may view or update all contracts of a certain category.
- Operating units are not used for security.

1.3.2.5 Oracle Contracts Integration (OKX)

A view layer between Oracle Contracts products and other products:

- Provides consistent view signature in cases where the source is dynamic (lines, rules, parties, contacts).
- Provides flexibility to change sources as need dictates.
- Not used over foundation components (AOL, JTF).

1.3.2.6 Extending Contracts Core

Shadow tables (one-to-one) are used when significant number of additional attributes needed by a module:

- Standard OKC forms call redirector API based on owning application.
- Module wrapper APIs call OKC APIs as needed.

1.3.3 Seed Data Protection

Seed data in Contracts applications can be customized, added, removed, altered, or extended for an installation. However, it is very important to protect some seed data from any changes. Different parts of Contracts applications and related integration with other applications may not work if seed data is violated.

To protect the integrity of an authored contract and integration to other applications the following objects have seed data protected:

- Line Styles
- Party Sources
- Role Sources
- Category

Access Levels

Seed data protection is categorized into three levels. These levels use the same approach as FND Lookups:

- System: Once added, the data cannot be updated or deleted. In a parent-child hierarchy, the addition of child level records cannot be done.
- Extensible: Once created, only new child records can be added in a parent-child hierarchy.
- User: All operations are possible.

1.3.3.1 Protection Levels for Objects

Line Styles

It is a requirement that Sales and Service Contracts protect seeded line styles. The Line Type is found in FND LOOKUPS and it is protected because it is defined with an access level of Extensible. Line Styles are also Extensible.

Tag	Protection Level	Operations Allowed
Line Style	System	Delete not allowed Changes not allowed to Line Styles Cannot add subline Cannot add/modify item source

Tag	Protection Level	Operations Allowed
Line Style	Extensible	Cannot delete existing line, subline/item source Can only add new sublines/item source Cannot expire existing item source
Line Style	User	Create, revise and update allowed on line styles

Role and Contact Sources

Party Roles are defined in FND Lookups with a Lookup Type as "OKC_ROLE - Contact Party Role". This information is fully protected because it is seeded as extensible.

An Access field with a LOV has been added for the Party Source and Contract Source tabs. Users can add role and contact sources with access level set as "User".

Tag	Protection Level	Operations Allowed
Party Source/Contact Source	System	Cannot delete/update existing Party Source/Contact Source. This protects addition of new Party Source/Contract Source
Party Source/Contact Source	User	Enter, delete/update existing Party Source/Contact Source

Subclass/Category

The protection set at the Sub-class level has the same impact on all the tags as well as sub-tags excluding responsibility tag.

Tag	Protection Level	Operations Allowed
Category/Header	System	Cannot delete a Category No changes allowed to any subtabs for example, cannot expire delete, or add

Tag	Protection Level	Operations Allowed
Category/Header	Extensible	Cannot delete an existing Category Can only add new Party Roles, Line Styles, and Responsibilities Cannot expire existing Roles, Line Styles, and Responsibilities
Category/Header	User	All operations allowed on Category as well as subtabs.

2

Contracts Integration Implementation Prerequisites

This chapter provides an overview of what needs to be accomplished prior to set up of Contracts Core. The chapter topics include:

- Mandatory Dependencies
- Required and Optional Modules
- Prerequisite Setup Tasks

Before proceeding with the implementation of Oracle Contracts Core and its dependencies, you should verify that you have installed Oracle Applications correctly. See *Installing Oracle Applications* for more information.

2.1 Mandatory Dependencies

Contracts Core depends on other Oracle modules and applications to extend functionality.

2.1.1 Oracle Contracts Integration

Contracts Integration is a layer of integration which allows Oracle Contracts Applications to interface to external Oracle applications such as Human Resource Management, General Ledger, Receivables, and so on. This layer of integration is transparent to the user and not functional. Contracts Integration is related to the views that support the Meta data (list of values for line items, customers, vendors, etc.). These views are seeded to retrieve data from other Oracle applications.

There is no setup for Contracts Integration. It is all seeded.

You can change or add new views in CRM Administration. Also you can change the source of lines, parties, contacts, etc. to point to a new view.

2.1.2 Oracle Application Library

Oracle Applications Object Library (AOL) is a required dependency for all Oracle applications. The AOL provides the following features for Contracts Core:

- System Administration
  - Set up users
  - Set up responsibilities for users
- Concurrent Processing
  - Programs
  - Requests
- iHelp Navigation Tree
- Lookups
- Profile Options
- Menu structure
- Workflow
  - Contract Approval Process
  - Notifications
- Flexfields
- Autonumbering Sequence

2.1.3 Oracle CRM Foundation

CRM Foundation suite of applications provides functionality for Contracts Core. The Foundation suite has been split into two groups:

- CRM Technology Foundation (JTT): Provides Java-based infrastructure software that is used to develop e-business solutions such as Sales, Marketing, and Contracts. It offers a common platform for developing applications with HTML, XML, and Java.
- CRM Application Foundation (JTA): Has many modules, however Contracts Core uses Resource Manager, Interaction History, and Task Manager.

2.1.4 Human Resource Management

To successfully run Contracts Core, you need to create an employee record, define organizations, and locations within the Human Resource Management System (HRMS) application.

2.2 Required and Optional Modules

The table below shows other modules integrated with Contracts Core. Some of integrations are optional. In some cases for those that are optional the workaround consideration is valid for those where data is pulled only.

For example, the LOV for customers comes from TCA (Receivables). Users can code their own PL/SQL view to pull data from some other table. The view needs to be registered using CRM Meta data. This then becomes available as sourcing for OKC. This applies to sources for rules, line styles, parties, and contacts.

Oracle Module	Mandatory/Optional	Valid Workaround
Receivables	Optional	Yes
Payables	Optional	Yes
Purchasing	Optional	Yes
General Ledger	Optional	No
Human Resource Management	Mandatory	N/A
Inventory	Optional	Yes
Pricing	Optional	No
Resource Manager	Mandatory	N/A
Interaction History	Mandatory	N/A

2.3 Prerequisite Setup Tasks

The following table shows the setup tasks, Oracle modules, and references for Contracts Core integration:

Setup Tasks	Oracle Module	Reference
Define persons Define locations Define organizations	Human Resource Management	*Managing People Using Oracle HRMS*
Define users Define responsibilities	System Administration	*Oracle Applications System Administrator's Guide*
Set up interactions	Interaction History	*Oracle CRM Application Foundation Implementation Guide*
Import resources	Resource Manager	*Oracle CRM Application Foundation Concepts and Procedures*
Define role types Define resource roles	Resource Manager	*Oracle CRM Application Foundation Implementation Guide*
Define resources	Resource Manager	*Oracle CRM Application Foundation Concepts and Procedures*
Set up Oracle Workflow	Oracle Workflow	*Oracle Workflow Guide*
Define customers Define accounting and invoice rules	Receivables	*Oracle Receivables User Guide*
Define suppliers Define payment terms	Payables	*Oracle Payables User Guide*
Define buyers	Purchasing	*Oracle Purchasing User Guide*
Define currencies Define daily rates Define conversion types	General Ledger	*Oracle General Ledger User Guide*

Setup Tasks	Oracle Module	Reference
Define UOM classes Define UOM Define item category Define master item Define item type codes Define category set Define organization items	Inventory	*Oracle Inventory User's Guide*
Define price lists Define discounts Define adjustments	Pricing	*Oracle Pricing User's Guide*

3

Implementing Contracts Core

This chapter provides the procedures for the setup and configuration tasks required to implement Contracts Core successfully. Depending on your business requirements you may not need to perform all the setup steps described here.

3.1 Setup Checklist

The table below shows all the setup steps for implementing Contracts Core. You can incrementally complete your setup, while starting to use Contracts. You may follow the order presented here. However, you can change and amend many of the setup steps while working with the product.

Step Title	Overview	Required
Setting up Security for a Category	Oracle Contracts is delivered with default categories that you can use. However, you have to make sure that the responsibility you are using has access to the category. By default, no responsibility has access to any category, unless explicitly assigned. Before you can access a contract of a specific category, you have to include the responsibility of your user in the category setup. See, Setting up a Category	Yes
Defining Inventory Item Types	To source line styles, you can use existing views, that are defined on specific inventory system item types. To use some of these integration views, you must define these item types first. See Defining Inventory Item Types	Yes
Scheduling Concurrent Programs	Schedule concurrent programs to control the automated status change and the scheduling and dequeueing of events components. See Scheduling Concurrent Programs	Yes

Step Title	Overview	Required
Setting up Article Set Codes	Define article sets to group articles. See Setting Up Lookup Codes	Optional
Setting up Subject Codes	Define Subject codes to classify articles. See Setting Up Lookup Codes	Optional
Defining Standard Articles	You can create and edit your own library of articles to meet your contractual requirements. To create and edit articles. See Defining the Library of Articles	Optional
Setting up Line Type Codes	The line type code defines the top line in the line style definition. See Setting Up Lookup Codes	Optional
Defining Line Styles	After you have defined the line type code, you can define the line style. See Defining Line Styles	Optional
Setting up New Rule Groups	You can create new rule groups in addition to the seeded groups. However, you cannot create new Rule Types. See Setting Up Lookup Codes	Optional
Modifying Rule Groups	There are seeded rule groups you may want to edit to meet your specific requirements. See Defining and Modifying Rule Groups	Optional
Setting up Party Roles	In addition to the seeded role like Customer or Vendor, you can add more roles such as OEM. See Setting Up Lookup Codes	Optional
Setting up Party Contacts	In addition to the seeded contacts, you can add more contact roles such as Arbiter or Legal Assistant. See Setting Up Lookup Codes	Optional
Defining Sources	The sources form allows you assign different sources for party roles and contact roles. You can modify existing source assignment or add sources to the roles you have defined in the previous two steps. See Defining Sources	Optional
Setting up a Category	You can modify seeded categories at any time. However, you should carefully plan your party definitions, rule groups, line styles and article definitions before defining additional categories. You can also copy a category. See Setting up a Category	Optional
Setting up Status and Operations	Define new statuses for a status type and define allowed operation for each combination of status type and category status. See Setting up Status and Operations	Optional
Setting up System Profile Options	For a list of all system profile options refer to, Setting up System Profile Options.	Optional

Step Title	Overview	Required
Setting up Contract Groups	Enables you to select public and private groups to place your contract. See Setting Up Contract Groups	Optional
Mapping Time Unit of Measure	You must map your own time units defined to the contracts internal time units seeded. See Mapping Time Units of Measure	Yes
Setting up Autonumbering	Automatic contract numbering can help classify a contract based on its attributes such as Business Group, Operating Unit, Class, and Category. See Setting Up Autonumbering.	Optional
Setting up XML Reports	This setup enables you to customize XML reports to show contract data. See Setting up XML Reports	Optional
Setting up Discoverer Reports	This reporting feature enables you to run the Contracts Expired but not Renewed and the Contracts Listing reports. See Setting Up Discoverer Reports	Optional
Setting up Contingent Event Codes	Contingent events may be used to schedule notifications. See Setting Up Lookup Codes	Optional
Setting up Termination Reason Codes	You can enter additional reasons (codes) for contract terminations. See Setting Up Lookup Codes	Optional
Setting up Status Change Reason Codes	You can enter additional reasons (codes) for status change. See Setting Up Lookup Codes	Optional
Defining a Process	If you want to add packages, procedures, or functions to be used in the Conditions form or Quality Assurance Checklists, you have to register them using the Process Definition form. If you want to add new workflows use this procedure. See Defining a Process	Optional
Defining a Quality Assurance Checklist	In addition to the seeded Default QA Checklist, you can define your own Quality Assurance Checklist. See Defining Quality Assurance Checklist	Optional
Setting up Actions	Actions allow you to define relevant changes of a contract to be used for event handling. See Setting up Actions	Optional
Defining Condition Templates	You can define conditions, that are independent of a contract, thus, conditions that evaluate for every contract. You can create templates for faster entry of conditions in contracts. See Defining Condition Templates	Optional

Step Title	Overview	Required
Setting up Pricing Integration	With this integration Contracts Core is able to price contracts using the various price adjustments at a time when they may be available for the items specified in the contract. See Setting Up Pricing Integration	Optional
Setting up Oracle Configurator	Through this integration a base model is entered in the Lines tab and using Oracle Configurator options can be selected and saved in Configurator Tables and Contracts Core table. See Integration with Oracle Configurator.	Optional
Setting up Notify Rules for Deliverables	You can set up a due date schedule and notification detail for a deliverable. See Setting Up Notify Rules for Deliverables.	Optional

3.2 Setting Up Notify Rules for Deliverables

You can set up a due date schedule and notification detail for a deliverable. Each deliverable schedule is a task and can be viewed in the Contract's Schedule tab or in Task Manager.

Steps

1. In the Authoring form, click the Rule tab.
2. Select the Notify Rule group.
3. In the Rule Detail region, enter a deliverable name (any name you want to assign to the deliverable, such as sending weekly status report).
4. Enter a due date for the deliverable. It could be due on a specific date, condition based, or a recurring schedule.
5. Enter Notification Lead Days which specifies sending a notification at the certain number of days before the deliverable is due.
6. Enter in Notify a person who should be notified that a deliverable is coming due.
7. Optionally, enter the Escalate Day which is the number of days after a deliverable is past due and needs to be escalated.
8. Optionally, in the Escalate To, enter the person who should be escalated to when a deliverable is past due.

9. Optionally, enter in the Escalate To (2) field a second person to be notified when a deliverable is past due.
10. Submit the Workflow Background Process concurrent program with item 'Contract Alert'. Select Yes for both Process Deferred and Process Time Out.
11. Submit the Send Task Notification concurrent program that sends a notification to remind someone that a deliverable is due. This notification will appear in the Launchpad (Inbox - click Find and select Deliverable Notification as notification type).
12. Optionally, Submit the First Escalation of Incomplete Task and Second Escalation of Incomplete Task concurrent programs if you have entered escalation rule details for first and second escalations.
13. Alternatively, you can run the Oracle Contract Tasks concurrent request set that includes requests for Send Task Notification and Escalation of Incomplete Task (Steps 11 and 12) and Action Assembler for Scheduled Planned Date.

Reference

- *To view deliverable schedule or close a completed deliverable, see Oracle Contracts Core User Guide, Executing Contract -Updating a schedule*
- *Oracle Contracts Core User Guide, Tracking Deliverables with Rules and Notify Rule*

3.3 Defining Inventory Item Types

Oracle Contracts allows you to use different sources for line styles. The underlying views for the linestyle sources have been defined. For example, if you want to add an item to the linestyle source "Consulting", you would have to create an inventory item with the inventory type "Consulting".

Currently the item types for the items that are to be used by the views have not been defined. You need to define inventory lookup codes to define items specifically for all of the following line styles:

Code	Meaning
CONSULTING	Consulting
SW LIC	Software License
K	Kit
MEDIA	Media Pack

Code	Meaning
TRAIN	Training
EDU	Education

Steps

1. From the Inventory responsibility, choose Setup > Items > Item Types. The ITEM TYPES Lookup window opens.
2. Enter a code.
3. Enter a meaning.
4. Save your changes and close the form.

3.4 Scheduling Concurrent Programs

There are several programs that need to be scheduled to run:

- Listener Programs
- Date Assembler
- Oracle Contracts Tasks
- Contract Status Change by Range
- Workflow Background Processes

The system administrator tools available for debugging problems related to queues are:

- Events Controller
- Asynchronous Errors

3.4.1 Listener Programs

Two concurrent programs need to be scheduled to run every xx minutes:

- Listener for Events Queue
- Listener for Outcome Queue

Use the concurrent the Oracle Contracts Core Listeners request set to start both programs. After submitting the Listener request set, each concurrent program should end with a normal completion status.

3.4.1.1 Event Listener and Outcome Listener Iterations

There are two profile options to set the work load of the listeners:

- OKC: Event Listener Iterations
- OKC: Outcome Listener Iterations.

These profiles control the number of messages the listeners will dequeue for every run. The profiles can be set at the site and application levels.

If the above profiles are not set, OKC Event and Outcome listeners will keep checking the queue messages all the times even if there is no message in the queue. This check is CPU extensive and uses memory resources.

If the above profiles are set, the listeners stop checking the queue if there are no messages and resume checking the next time listeners are scheduled to run (based on the schedule parameter you set when running the concurrent request for Event and Outcome Listeners).

The default load value is set to 100 (recommended). This number can be changed to suit the Advanced Queue load. The minimum should be greater than 1 (otherwise there is no point in running the process) and the maximum should not exceed 1,000.

This number should be set to a higher value for systems that have larger work loads. Setting a higher value for the profile options, results in longer running and more resource (CPU) intensive listener processes.

Submit requests to start both listeners using a "Periodic" schedule. The suggested period is every 3 to 5 minutes. Failure to run the Listeners on a regular schedule will severely limit functionality in Contracts and other applications using the Contracts Events System. It will also cause Events system Advanced Queue to become back logged.

Listeners run and complete rather than running indefinitely. In order to process messages in the queue, you must set the listeners to run on a regular schedule. Increase or decrease the interval/period at which the Listener process run according to your system load. The sleep parameter seen when submitting a request is now redundant.

You should balance these processes with the iterations and request schedule. Use both these methods to fine tune your listener processes.

3.4.2 Date Assembler

The Date Assembler concurrent program should run once every day, preferably later in the evening. This program is used for tasks such as automatically renewing contracts that are eligible according to the contract definition.

3.4.3 Oracle Contracts Tasks

To process Notifications, the request set "Oracle Contracts Tasks" needs to be started. This includes four programs:

- Send Task Notifications
- First Escalation of Incomplete Tasks
- Second Escalation of Incomplete Tasks
- Action Assembler for Scheduled Planned Date

3.4.4 Contract Status Change by Range

This concurrent program updates the status of contracts such as: expiring contracts, realizing a future dated termination or activating a signed contract. You should schedule this concurrent program very early in the morning, so that the contract statuses are always synchronized with their effective dates. See Also Status and Operations.

3.4.5 Workflow Background Processes

When using Oracle Workflow with other applications, the program "Workflow Background Processes" most likely is started to run in recurring intervals. If the "Workflow Background Processes" is not scheduled to run, the system administrator must start this process to run regularly (i.e. to run every five minutes).

Steps

1. From the System Administrator responsibility, select Concurrent > Requests.
2. In the Name field, select the Workflow Background Process option.
3. Select Contract Alert from the LOV for Item.
4. Select Yes from the LOV for Process Deferred.
5. Select Yes from the LOV for Process Time-out.

6. Click OK.
7. Click Schedule.
8. Select Periodically for Run the Job.
9. Enter 5 (this is determined by the system administrator) and select Minutes from the LOV for Rerun every.
10. Click OK.

3.5 Using the Events Controller

The Events Controller is a tool for the system administrator to debug any environment problems related to the queues. This is a query only form which gives the system administrator the provision to check common problems related to advanced queues, for example whether the listeners are running, queue objects are valid, or if there are any errors thrown by the queues. This form provides a more convenient method of viewing more information from one location.

The Events Controller has a start and stop option for queues and concurrent programs. The queues must always be enabled (started). The listeners will be running as scheduled and will show an error if the queues are disabled (stopped). The start and stop option is an extra DBA provision provided to the system administrator and can also be used when debugging certain queue problems.

If the Events Controller is used to stop queues, it will automatically terminate any Listener for Events/Outcome concurrent programs. You can start the queues again and restart the listeners. This will not cause any data loss from the queues.

Note: The Asynchronous Errors form only displays queue errors. For more information see, the next procedure *Reviewing Errors from Asynchronous Processes*.

This form contains three tabs:

- Background Process
- Queue Content
- Queue Status

To locate this form from Contracts Manager responsibility, navigate to > Contract Events > Events Controller.

3.5.1 Background Process

This tab displays the events and outcome listeners that are running in the background and the workflow engine process. In the event of an error, this tab provides you with the ability to access the Errors window which describes the error and the cause.

The following attributes are displayed in this tab:

Attribute	Description
Request Id	This is the Id of the concurrent program submitted.
Listener/Program	Identifies the name of the event, outcome, or workflow background process.
Phase	The Phase displays Pending, Running, or Completed.
Status	The Status displays Normal or Error. When a program has been completed with an "Error" status, you can click Diagnostics to view the error details returned by the concurrent program.
Request Date	Identifies the date when the program was submitted.
Requestor	Identifies the user who submitted the program.

3.5.2 Queue Content

This tab displays the Queue Messages and Queue Errors that have occurred. If an event is launched (such as contract signed) you can expect to see the queue content within this tab. If there is nothing in the queue, then the Queue Status tab is used to check the queue status and also to check if all queue objects are valid.

The following attributes are displayed in this tab:

Attribute	Description
Queue Name	Identifies the name of the advanced queue.
Correlation	Identifies the correlation Id such as KSIGN and KEXTEND.
Consumer	Consumer is the program which parses and consumes the message in the queue.
Enqueue Time	Shows date when the message was placed in queue.
Queue Content	Displays queue content.
Source Name	Shows source of error.

Attribute	Description
Retrys	Displays number of times the queue tried to dequeue the message.
Begin Date	Shows Date when error occurred.
Error Details (Button)	Enables you to view the details of queue errors.

3.5.3 Queue Status

This tab displays the queue names, queue objects, and their status (valid or invalid).

The following attributes are displayed in this tab:

Attribute	Description
Queue Table	Shows the queue table name.
Queue Type	Displays the queue type such as Normal and Exception.
Enqueue Enabled	Shows either Yes/No
Dequeue Enabled	Shows either Yes/No
Start Queue/Stop Queue (Buttons)	Enables you to start or stop the queue.
Queue Objects	Names all the OKC advance queue objects.
Object Type	Displays type of object such as Table and View.
Status Type	Shows whether object is valid or invalid.
Last Modified	Displays date when the object was last modified.
Details (Button)	Shows details for rules and subscriber objects.

3.5.4 Reviewing Errors from Asynchronous Processes

The Events Component is dependent on asynchronous processing. It uses Advanced Queuing System to achieve this. This system is totally transparent to the user. The messages being processed may encounter exceptions and as a result the messages are rolled back and will be reprocessed again at a specified time. There is a need to record any exceptions encountered. Hence whenever an exception is encountered, the Evaluator writes the details of the exceptions to a table OKC_ AQERRORS and the error stack to OKC_AQMSGSTACKS tables.

The Asynchronous Errors window displays all error messages of advanced queueing processes that have failed. This window displays Queue Name, Message ID, and the Message Text. Usually this window is used by your support representative to resolve issues. To access this window, from the Contracts Manager responsibility choose, Contract Events > Query Asynchronous Error Messages.

3.6 Defining the Library of Articles

The Library of Articles is a small database of previously written and established articles. These articles can be referenced and included in a contract. This topic group consists of the following:

- Article Comparison
- Checking Setup for JSP
- Creating a New Article
- Editing an Article
- Modifying an Article Set

3.6.1 Article Comparison

This feature enables you to compare the difference between the original text and the varied text of a standard article when you click the Compare button. A browser opens showing the varied text that is added, removed and unchanged:

- Text in a different text color (such as red) indicates additions from release of a later creation date.
- Text removed from an earlier release is indicated by ~~strikethrough~~.
- Text in black means the text is unchanged from earlier creation date.

This feature is helpful in identifying changes to an article during the drafting or editing stages. There are two forms which enable you to access the article comparison feature:

- Standard Articles Library form (Navigate to Setup > Contract > Standard Articles > Define Articles)
- Article Text form (Navigate to Contracts Authoring > Articles > Show Text)

You can make changes to the saved varied text as many times as needed. Article Comparison only compares the latest varied text with the standard article.

Comparison Example

Release 1.7

If not otherwise specified on the Order Form, this Agreement and each program license granted under this Agreement shall continue perpetually unless terminated under this Article IV as of December 31, 2002.

Release 1.8

This Agreement and each program license granted under this Agreement shall continue perpetually unless terminated under this Article IV as of December 31, 2005. The Vendor shall deliver to the customer location for use in the territory 5 copies of the software media (Master Copy) for each program currently available in production.

Comparison Results:

~~If not otherwise specified on the Order Form, this~~ This Agreement and each program license granted under this Agreement shall continue perpetually unless terminated under this Article IV as of December 31, ~~2002.~~ 2005. The Vendor shall deliver to the customer location for use in the territory 5 copies of the software media (Master Copy) for each program currently available in production

3.6.2 Checking Setup for JSP

The article comparison feature is a JSP page that is invoked by clicking the Compare button from either the Contracts Core Standard Articles Library form or from the Article Text window within the Contracts Authoring Articles tab. Therefore the JSP page is invoked with the same responsibility and user Id used to log on to Contracts Core. The settings should be checked when the page does not appear.

References

- *Oracle System Administrator's Guide*
- *Oracle Applications Developer's Guide*

Steps

Check Profile Option Values

The values can be defined at User, Responsibility, Application, or Site level. The application is Oracle Applications Shared Technology.

1. From the System Administrator responsibility, navigate to Profile > System. The Find System Profile Values window opens.

2. Query the following profiles shown in the table below and modify as necessary:

Profile	Description	Example Value
Applications Web Agent	The value should be the URL for the PL/SQL Agent.	*http://ap103jvm.us.oracle.com:5801/pls/kdv7core*
Apps Servlet Agent	The value should be the URL for invoking the JSP page.	*http://ap103jvm.us.oracle.com:5801/oa_servlets*
ICX: HTML Directory	The value should be the alias name used for the HTML directory.	*OA_HTML*

Check Function

1. From the Application Developer responsibility, navigate to Application > Function. The Form Functions window opens.
2. Query "OKC_CONTRACT_ARTICLE_DIFF" for Function.
3. Click the Properties tab.
4. View the following value for Type: "SSWA JSP FUNCTION".
5. Click the Web HTML tab.
6. View the following value for HTML Call: "okcContractArticleDiff.jsp".
7. Close the form and save any changes.

3.6.3 Creating a New Article

Prerequisites

Lookups must be defined for article sets (OKC_ARTICLE_SET). From the Contracts Manager responsibility, choose Setup > Contract > Standard Articles > Define Set.

Lookups must also be defined for article subjects (OKC_SUBJECT). From the Contracts Manager responsibility, choose Setup > Contract > Standard Articles > Define Subject.

Steps

1. From the Contracts Manager responsibility, choose Setup > Contract > Standard Articles > Define Articles. The Standard Articles Library window opens.
2. Select an Article Set, and navigate to Actions > New Article. The new article will automatically be placed in the selected Article Set after entering name and subject.
3. Enter the Article Name.
4. Select Subject from LOV.
5. From the Releases tab, enter a Release Number. If you are either entering the first release for a new article or entering a new release for an existing article, you need to include the Release, Active Date (or accept default), and optionally a Description.
6. Click Release Text button. The Release Text window opens.
7. Enter the text for the article.
8. Click OK.
9. If this article is incompatible to another article, select the Incompatibilities tab.
10. Select an Article Name for Incompatibilities.
11. From the Article Sets tab, select a name from the LOV for Member of Article Sets.
12. Save your work.
13. Select the Article Set from the tree where you placed your article. Your article title appears.

3.6.4 Editing an Article

Steps

1. From the Contracts Manager responsibility, choose Setup > Contract > Standard Articles > Define Articles. The Standard Articles Library window displays a tree of existing article sets and the articles within each set.
1. If you want to edit or review an existing article, then select an article in the Article Set.
2. Select the Releases tab.

3. If you want to find contracts where the article appears, click Used in Contracts. The Used in Contracts window appears. This window shows the Contract Number, Contract Number Modifier, and a Short Description.
4. Click OK to close the Used in Contracts window.
5. Enter the Release, Date Active, and optionally a Description.
6. Click Release Text. The Release Text window appears.
7. Enter your changes.
8. Click OK.
9. Click the Select check box (found on right side) to enable you to compare this article release.
10. Click the Select check box of another article release for comparison.
11. Click Compare. The Standard Article Release Comparison window appears.
12. View the comparison results.
13. Close the Standard Article Release Comparison window.
14. Make any additional edits as necessary.
15. Close the Standard Articles window.

3.6.5 Modifying an Article Set

Steps

1. From the Contracts Manager responsibility, choose Setup > Contract > Standard Articles > Define Articles. The Standard Articles Library window displays a tree of existing article sets and the articles within each set.
2. Select an article set from the tree.
3. If you want to associate or modify Price Types to an article set, click the Price Types tab.
4. Select a Price Type (Time and Material or Firm Fixed) from the LOV.
5. If you want to update the category the article is associated to click the Categories tab.
   - To add a category, choose a category from the LOV.

- To remove a category, select the category from the Category column and navigate to Edit > Delete.

6. If you want to associate Rule Groups with an article set, click the Rules tab.
7. Select a Rule Group from the LOV.
8. Save your work.

Guidelines

Creating new releases of articles is a way to maintain different versions of an article. Creating a new release does not impact articles in other contracts. To change an article already in use, you must create a new release, which automatically becomes the most current version after the previous release expires.

Articles can be defined to be incompatible with each other. For example, you could make a payment term of 30 days net incompatible with a 10% advance payment. You cannot define an incompatibility before both incompatible articles are created.

If you select an Article Set, then the tabs change to Price Types, Categories, and Rules. In the Price Types tab, you can choose price types for your article set. When you author a contract, then you can reduce the article sets available by Price Type.

When you assign a category to a standard article set, then this article set appears in the list of values when you author a contract that is assigned the category. You can then drill down to articles from that set. However, the Authoring window does not limit the articles you select to sets that are assigned to a category. You can still access articles directly when authoring, even though their article set is not assigned to the current category. See also:

Setting Up Lookup Codes

3.7 Defining Line Styles

A contract is composed of lines of text. Control the type of information that can be entered on a particular line by defining a line style. The line style sets input requirements and sets up the lists of values to choose from in a contract line during contract authoring.

The most general and least detailed line in a contract is called the top line. Oracle Contracts Core delivers a set of line styles you can use, or you can create new line styles by following the delivered set as a guideline. You can add line styles in a hierarchy below the top line style to control detailed information within the top line.

References

Oracle Contracts Core User Guide, Line Style Overview

Prerequisites

Top Line styles must be defined in lookup codes (OKC_LINE_TYPE). From the Contracts Manager responsibility choose, Setup > Contract > Categories and Sources > Define Line Types.

Steps

1. From the Contracts Manager responsibility, choose Setup > Contract > Categories and Sources > Define Line Styles. The Line Styles window displays existing line styles in a tree format.
2. If you want to add a line style below an existing line style in the hierarchy, then select the parent line style.
3. Click the **New** icon in the Menu bar. This is the button with a green plus sign. A new node will appear in the line style tree with a generated name.
4. Enter a line style name.
5. Choose a line type to classify your line style.
6. If the line style contains a priced item, then select Priced. The tree structure to the left will mark priced items by displaying an arrow.
7. Choose one or more sources for the line style information and enter start and end dates.
8. Save your work. The line style is saved as part of the line style hierarchy.

Guidelines

There is one special line type: “Free Format”. This line type allows you to enter ad hoc definitions that do not rely on existing sources like inventory items.

You can create a price line style at any level in the line style hierarchy. However, you can only price one line in the hierarchy chain.

For all line styles, you can determine the source of the line style. In the Authoring window, lines with the price flag disabled will not show a list of values if you select system items (Inventory) as the source. If a system item line style is not priced, then the item source list of values is retrieved in the Item sub tab of the line.

If a line style is for a top line for a service contract type, then the Valid Operations tab appears. You can see whether or not entitlement and invoicing are allowed for the line styles delivered for Oracle Services.

Entitlement and Invoicing is used for Oracle Contracts for Service only.

3.7.1 Pricing Setup for Line Styles Overview

Line styles defined in Oracle Contracts will play a key role in mapping between Contracts data into the format which Oracle Pricing understands to qualify and calculate various prices and adjustments for an item.

In Oracle Contracts, lines based on these line style are the ones that hold the items specified in a contract as well the special attributes corresponding to these items. These lines will be used to construct price request lines to be sent to Oracle Pricing.

A request line in Oracle Pricing is a line which holds an item and all its attributes together. The item on the request line is priced by the Pricing engine while using its attributes as qualifiers for various pricelists and modifiers. For example, the following line styles hierarchy.

Service

- Product
  - Deployment

This group of line styles is used to store information about service on various products and price of the service could vary based on where the product is deployed and price for service is some percentage of the price of the product.

So the lines corresponding to this line style group would look something like this-

Gold Service

- Movie Camera
  - USA
  - INDIA

Then, in this case the price of Movie Camera would need to be determined and then based on the price you can calculate the price of Service for the camera. Also, you may want to know what would be the service charge if the Camera is deployed in the USA.

Two price request lines are sent to the Pricing engine:

- Price of Movie Camera deployed in the USA.

- Price of Gold Service for the Movie Camera deployed in the USA.

Using this information, the Pricing engine will return the prices for line one and Line two.

To construct these two request lines, you need to know which linestyle is holding the item and at which level to price and if there is any relationship between two items listed. To capture this information, the Contract administrator will select these check boxes for line styles:

- Priced_item or Item_to_price (PI): The item that you want to price.
- Basis_For_Priced_Item (BPI): (Reserved for future functionality)
- Priced (P): The lowest level at which you would like to price PI or BPI.

In the example line style group, the check box selection is as follows:

Service (PI)

- Product (BPI)
  - Deployment (P)

The Priced (P) flag indicates the lowest denominator in the list of attributes for a Priced Item (PI), based on which the price of the priced item (PI) can vary. In the authoring form, the lines corresponding to priced (P) line styles will be the one holding the price for the priced item(PI).

3.8 Defining and Modifying Rule Groups

A rule group is a collection of one or many rules. Rules collect one or many related pieces of information. Rule groups control which rules are used in a contract.

Your organization can assemble rule groups to enforce business processes and company standards. You create rule groups so that rules with similar characteristics can be easily referenced and accessed. For example, create a rule group for billing that would group together rules for bill to address, billing schedule, and payment terms. Some rules in the group can be defined as optional.

You can use rule groups to make certain rules optional entries for one rule group, while leaving it a mandatory rule for another rule group. Assign rule groups to your contracts according to your business procedures. You can save incomplete data, but all non optional rules have to be complete to pass Quality Assurance.

Prerequisites

A rule group must be defined as a lookup (OKC_RULE_GROUP_DEF). From the Contracts Manager responsibility choose, Setup > Contract > Rules > Define Rule Groups.

Steps

1. Open the Rule Group Definition window. From the Contracts Manager responsibility, choose: Setup > Contracts > Rules > Assign Rules to Rule Groups.
2. If you want to edit a rule group, then select the rule group.
3. In the Name field, you can add or change rule types.
4. Select the Optional check box if you want to set the rule type as optional.
5. Min and Max Occurrences fields determine the minimum and maximum occurrence of instances of a rule within a rule group.

   For example, if the min =1 and max=2 for the Notification Schedule rule (which is a rule type) of Notify rule group. In the Rule tab of Contract Authoring form, select Notify rule group. The Notification Schedule is displayed in the Rule region. Since maximum occurrences is set to 2, this means you can add another notification schedule by clicking the + icon.
6. Select the Pricing check box if the rule can be an attribute to determine the price in a contract. For example, the price of a line can be based on the rule Ship To location while a Notification rule may not be used for this purpose.
7. Use the Rule Type region to add or change rules for each rule group. Rule Type Source describes the source the rule is associated with.
8. In the Source field, select a rule source type from the list of values. If the rule is based on JTF objects, the source should be defined.
9. Use the Object Number field to indicate the data base column that should be used to store that particular source. The Object Number identifies which column to be used if one rule uses multiple external sources (JTF objects). OKC_RULES supports three JTF objects in one record. They are:
   - OBJECT1_ID1
   - OBJECT2_ID1
   - OBJECT3_ID1
10. Save your work.

Guidelines

When you select a rule group in your contract, all the rules in that group are automatically copied into your contract. Each rule may be optional or mandatory.

To access a rule from the Authoring window, you must include the rule group in the category definition. You can include more than one rule group to a category.

Only **Seeded Rules** are supported. Users are **not** supposed to create their own segments in OKC Rule Developer DF. Even if you are able to set new rules and make it work, the subsequent patches could remove them.

3.9 Defining Sources

Use this procedure to determine the different lists of values that will appear as sources for your roles. For example, in a contract where you are buying services, you the customer should show in the list of values for the role of the customer, though you have not defined yourself as a customer in your Oracle Receivables customer list. In this case, you define the customer for a buy contract to select an organization.

Steps

1. From the Contracts Manager responsibility, choose Setup > Contract > Categories and Sources > Define Role Sources. The Role Sources window opens.
2. Select a party role.
3. In the Party Sources tab, select a source for the party, select buy or sell, and enter effective dates.
4. In the Contact Sources tab, click New from the Menu bar to add a new contact. A new row is added.
5. Select a contact, a source for the contact, select buy or sell, and enter effective dates. The constrained flag indicates that a relation from the contact to the party is enforced through the party ID.
6. Save your work.

Guidelines

You cannot delete a source from a role, once you have defined it. To change your source assignment, you expire the current source or contact source assignment and add the new source assignment with a start date following the expiration date of the prior assignment.

The sources you can choose from are predefined objects.

3.10 Setting up a Category

A category is a contract type associated to a contract class. A contract class determines the authoring form used. For example, when the class is "Corporate" the Contracts Core authoring form is used. Examples of other classes include:

- Contract for Sales
- License Agreement
- Master Agreement
- Rights Contract
- Service Agreement

Contract categories are seeded. You can edit seeded categories and create additional categories as needed. However, some Oracle Contracts modules, such as Oracle Service Contracts, do not support creating additional categories.

When you create or modify a category, the Authoring window becomes tailored to suit the category by selecting components from:

- Access Level
- Opportunity Creation
- Party Roles
- Rule Groups
- Line Styles
- Responsibilities

Follow the steps listed below for creating or modifying a category.

Prerequisites

Responsibilities must be given access to a category before they can read, create, or modify contracts. For example when you author a contract or when you create a new contract using the Copy function, the responsibility must have Modify access to the new contract category. Responsibilities are defined in System Administrator.

Steps

1. From the Contracts Manager responsibility, choose Setup > Contract > Categories and Sources > Define Categories.
2. If you are creating a new category, enter a name.

> **Note:** The Category Code field is a system generated value that displays <Class Code> n (where n is a number) after the record is saved. This feature ensures duplicate categories cannot be created across contract classes.

3. Select Start Date (or accept default).
4. Select an Access Level.
5. Select a Class.
6. Enter a Description.
7. Enter an End Date.
8. Select the Create Opportunity check box, if this category will be used for opportunity creation.
9. From the Party Roles tab, select roles that can be included in the category and effective dates.
10. From the Rules tab, select the Rule Groups you want to access from any of the three levels: contract, line, or articles.
11. Select the Party Roles from the Rule Party Roles region.
12. Qualify the Party Roles by selecting either Object or Subject for the Mode. This maybe needed when there are multiple vendors or customers in a contract, but you only want to pay or invoice one of them. Select the role, even if your contract has only one vendor or customer.
13. Select the Optional check box if the qualified role may be omitted for a rule to pass the Quality Assurance.
14. From the Line Styles tab, select top line styles. The tree structure of the entire selected line style is displayed. The Party Roles and Rule Groups previously selected appear within the subtabs.
15. If you want a party role to be selectable on the contract line level, choose the Select check box from the Party Roles subtab.

16. If you want a rule group to be selectable on the contract line level, choose the Select check box from the Rules subtab.
17. From the Responsibilities tab, select at least one responsibility. A responsibility that is not selected on the Responsibilities tab cannot access to the contract.
18. Select an Access Level.
19. Select effective dates.
20. Close the form and save your work.

Guidelines

You cannot delete a component from a category. To remove a component from the category, you have to set it to expire. Adding components to a category do not make the components mandatory during contract authoring.

Rule Party Roles are used for Oracle Core and Service Contracts. The following is an example of qualifying a role for a rule:

> Qualify the customer role in the billing rule as the object (the party to which the bill is sent). Also you can set your billing rules to identify the seller as the subject (sending the invoice).

3.10.1 Copying Contract Category

You can create new contract categories using the Copy Contract Category feature. This feature enables you to create new categories for contract classes such as Contract for Sales.

It also provides you with the ability to create new categories for other contract classes such as Corporate, License Agreement, and Master Agreement. The new category takes on the same access level as the copied category, so for example when copying the category Contracts for Sales, which is of class Contract for Sales, the new category will have the same access level as the Contract for Sales category, preventing new line styles and party roles from being created or modified.

Steps:

1. From the Contracts responsibility, navigate to Setup > Contract > Categories and Sources > Define Categories. The Define Categories window opens.
2. Enter the query mode, and choose a class, such as Contract for Sales.
3. Run the query.

4. From the Actions drop down menu, choose Copy. The Copy Category window opens. The Source Category field will default in based on the source of the copy.
5. Enter a name for the new category.
6. Enter a description.
7. Click Copy. A new contract category is created, taking on the same attributes of the copied category.
8. Save the new category.

3.11 Contract Access Considerations

You can fine tune access to contracts, such as granting a specific user rights to see a contract, but not to modify it. The category definition includes granting access levels to responsibilities. For each category you can define whether a given responsibility has Read-Only or Modify access.

Access to contracts involves the following:

- Contract Group Access
- Allow Operations
- Multi-Organization Access
- Restricted Operations
- Troubleshooting Contract Access

3.11.1 Contract Group Access

Your first level of access to contracts is via the contract groups listed in the Contract Navigation window. Although you have access to a contract group, you do not necessarily see all the contracts in the contract group because other levels of security may be in place.

You can assign one or more contract groups to each contract. There are two kinds of contract groups: public and private. A private group is created by a user and only the user creating the group has access to that group. A public group is available to all users.

Public groups can only be created by users that have the OKC Public Group Creator system profile option enabled. If this system profile option is not set or is disabled, then the user can only create private contract groups.

Everybody with access to a contract group who is assigned to the contract group has full read and write access to the contract, as defined in the Status and Operations window.

Use contract groups to navigate to contracts through different navigation paths. A contract is referenced in contract groups, not copied. Assign contract groups in the Details subtab of the Summary tab of Authoring window. As a contract administrator, you can use the Contracts Groupings window to assign and unassign contracts to groups.

3.11.2 Allow Operations

The Status and Operations window defines, by category and status, which operations can be executed against a contract. The allowed operations are defined by status and not by status type.

In other words, if you define a new status for the status type Active, then you have to make sure that you specifically allow operations such as on-line update. If you create a new status without specifying any allowed operations, you implicitly allow no operations for this contract status.

3.11.3 Multi-Organization Access

When you log on to Oracle Applications, you automatically sign on within your organization. By default, your actions are restricted to read and modify contracts for your organization only. For some business uses, where contracts need to be accessed 24 hours a day from any organization around the world, contracts need to be accessible outside your organization.

If you need to give access to contracts across organizations, you can use the Administration subtab in the Summary tab of the Authoring window. You grant Read-Only or Modify access to resources independent of the organization the user is in.

To create a resource, you must choose the CRM Resource Manager responsibility. A resource in this context refers to the resource definition of CRM, for example, customers, suppliers, and employees.

If you want to use persons or roles already declared somewhere else in the application setup, for example an employee, then you must import these sources using the Import Resources function in the CRM Resource Manager responsibility. Make sure you assign contract execution rights in the Define Resource menu.

Note: Make sure that you have defined the role completely before importing the resource, for example: Define the employee and tie that employee to FND_USER, before you import the resource. You can import only once.

3.11.4 Restricted Operations

Although you have modify access to a contract via your responsibility or access group, you may still be restricted from performing certain operations against the contract. You can only perform operations that are specified for the status.

If your organization uses changes requests to update a contract and another user is in the process of applying a change request, then you will not be allowed to update the contract. You can submit another change request while another change request is being applied, but you must wait until the prior change request is completed before you can apply your change request to the contract.

There are also restrictions for changing the status of a contract. For example, you cannot terminate a contract that was never signed, instead it has to be cancelled. You cannot cancel a contract that is active, it can only be terminated. See Status and Operations.

3.11.5 Troubleshooting Contract Access

User does not have access to contract group.

- Check if contract group is public or private.

User cannot see the contract in the contract group.

- Check if the user is in the same organization and has access granted.
- Check if the user is using a responsibility with access to the category.

User can see the contract, but cannot update the contract.

- User was not given access.
- Responsibility with modify access is not assigned to the contract category (in the Define Category form)
- User is using a responsibility with read-only privilege instead of modify.
- The Operations window excludes that operation or this operation is not included, thus implicitly disallowing that operation.
- The contract may be in a status change that does not allow any operations.

3.12 Setting Up Status and Operations

You can control the operations (such as update on line, update via change request, and delete) that can be performed on a contract depending on the contract category and the status of the contract (such as active or terminated).

The status of a contract is a label identifying where the contract stands in its life cycle. Examples of status types include:

- Entered: The contract is currently being edited and it can be completed but not approved.
- Signed: The contract is approved, but not yet effective. This status is used when the contract is not yet due, but has the same protection from changes as an approved contract.
- Active: The contract is approved, signed, and effective.
- Hold: The contract was set on hold from signed, active, or another hold type. For example, use the Hold status when a customer is moving or the contract is in dispute.
- Expired: The contract was active, but is not effective anymore.
- Terminated: The contract was active, but was terminated by either party before the contract expired.
- Canceled: the contract never was active and is not planned to become active.

You can define as many statuses per status type as you need. You can have a different status on the contract header and on the contract lines. For example, you can terminate a single contract line, while the rest of the contract is still active.

You can change the status of a contract:

- Manually
- Through the approval process
- Through the extension process
- Through the termination process
- Automatically by the Status Change concurrent program

From the Launchpad you can initiate a manual status change. Some status changes have to be done through a concurrent program. For example, the Status Change program automatically changes a Signed contract to Active when the start date is reached.

Custom Status

To provide greater control of contract status, you can add custom statuses for any status type. For example, to distinguish between entered contracts that are draft and those pending approval, you can define draft and pending statuses for the entered status type. For each status type, you define the default status that is automatically assigned when a status type automatically changes.

Operations

You can control the operations allowed per contract status and category. For each user status, you can define the operations that are supported. For example, you can specify that draft documents can be updated on-line, but contracts pending approval must go through a change request process. You can also state that once a contract becomes active it cannot be updated at all.

Allowed operations are defined by status, and not by status type. If you define a new status for the status type Active, then you must also specifically allow operations for the new status, such as on-line update. If you create a new status without specifying any allowed operations, then that status allows read-only access.

Adding Statuses

Use this procedure to add statuses and define, by category and status, the operations that can be executed against a contract.

Steps

1. From the Contracts Manager responsibility, choose Setup > Contract > Status and Operations. The Status and Operations window opens.
2. Select a status type.
3. Optionally, enter additional statuses for the status type.
4. Enter the text to display in application windows in the Meanings field.
5. Select the Default check box for the status of a contract when it first reaches the stage of the selected status type.
6. Select every category, operation, and level (header or line level) combination you want to relate to the status.
7. Select the Allowed check box in the Allowed Operations by Category region to allow the operation for each line you created. Clear the Allowed check box to prohibit the operation.
8. Close the form and save your work.

Guidelines

Statuses are seeded for Status Types within the Contracts Core application, however for Sales Contracts Status Types need to be set up.

If you define a new status for the status type Active, then you have to make sure that you specifically allow operations such as on-line update. If you create a new status without specifying any allowed operations, then you implicitly allow no operations for this contract status.

For the Status Change concurrent program to automatically update contract status, you must define a default status for each status type.

3.13 Setting Up Contract Groups

A contracts group provides the ability to store a collection of contracts together that may share a given characteristic. The number of contract groups you need is depend on your business requirements.

References

Oracle Contracts Core User Guide, Maintaining Contract Groups

3.14 Mapping Time Units of Measure

Oracle Contracts Core defines the Unit of Measure (UOM) conversion for time differently from Oracle Applications. This helps to ensure that the scheduling is more accurate than a simple conversion. For example, 1 month = 30 days, which is only correct for 5 out of 12 months of a year.

If you want to define your own time unit conversions for extending a contract or for scheduling, then you must define your own time unit conversions. For example, lets say you want to define the 4 time units in the Units of Measure menu: Day, Month, Quarter, and Year. The UOM Class is Time. The OKC: Time UOM Class system profile option should be set to the value of Time (Note: This is case sensitive and "Time" should be entered using the exact case). When you use the Map Time Units window, the list of values will only show the time units you assigned to the conversion class Time. When you map your own time units to the internal time units:

- Day: 1 Day
- Month: 1 Month
- Quarter: 3 Months

- Year: 1 Year

With this setup, you can select only the values listed above from the LOV when defining time components. You will not have access to other time units such as hour or minute.

Prerequisites

Define your units of measure classes and units of measure using Oracle Inventory.

Choose the unit of measure class to be used for Contracts, and select it for the OKC: Time UOM Class profile option. The units of measure within the chosen class appear in the User Unit field list of values in the Map Time Units window.

Steps

1. From the Contracts Manager responsibility, choose Setup > Contract > Units of Measure > Time Units of Measure. The Map Time Units window opens.
2. Select a user unit of measure from the list of units of measure.
3. Select the base unit of measure that equals the user unit of measure.
4. If needed, enter conversion information.
5. Optionally, enter a description.
6. Save your work.

Guidelines

There are six internal time units: minutes, hours, days, weeks, months, and years. Make sure to map each time unit you want to use in Oracle Contracts Core. Example of a mapping: Day (your definition) = 1 day (base definition).

3.15 Autonumbering Contracts

Autonumbering Contracts consists of the following topics:

- Setting Up User Function

3.15.1 Overview

Autonumbering functionality automatically populates the contract number upon initiation of a new contract. Utilizing the Autonumbering functionality, you will have a greater flexibility in defining how contracts are numbered.

Contract numbers can be either sequential numbers or a combination of defined prefix and suffix alpha-numeric characters to classify a contract based on its attributes. These attributes include:

- Site
- Business Group
- Operating Unit
- Class
- Category

You can set up auto-number classification using a prefix and suffix with a contract number. For example, a Service contract originating in the United States can be numbered US-9999-SRV. Use of a prefix and suffix is optional. The numeric part of the contract number will be formatted to the length specified for that sequence. For example number 99 will be formatted as 00099 if the length is specified as 5 digits.

Autonumbering of contracts will be helpful in a scenario when contracts are entered or imported from an external source or system, built from another document, or when contract data is entered manually. From the Contracts Launchpad, a contract number can be automatically generated when you initiate a new contract by:

- Launching Contracts and selecting New from the Tools drop-down menu.
- Navigating to the Contract Navigator tab, selecting a contract group from the list, highlighting a contract from the right column, and either right clicking the mouse and selecting Copy or selecting Copy from the Tools drop-down menu. In the Copy form, the New Contract check box must be selected.
- After creating and saving a quote, the user can navigate from the Order Capture form to Actions > Create Contract. This initiates a new contract with an automatically generated number.

You can manually override the system generated number, assuming the Allow Manual check box is selected during setup. In addition, if there is no setup defined,

manual numbering is assumed and the Contract Number field is not protected from update.

When a contract is created as a result of running a concurrent program, for example creating a service contract using the Service Contracts Order Capture Integration concurrent program, the contract number is automatically generated even if the Sequential Numbering profile option is not set up. That is the profile option is set to 'Always Used' at the session level when a contract is created from a concurrent program.

If a contract is created online from other applications, for example creating a contract from a quote in the Quoting form, any product integrating with Core Contracts should set this profile option at the Application level for that product to Always Used or Partially Used in order to use the Autonumber Sequencing setup in Core Contracts.

3.15.2 Autonumbering Features

The Autonumbering form is used for entering contract numbering setup. In the top section, you decide how to number the contracts. There is only one header level record for the entire installation. In case of a user-defined function, there will be no details required and the Sequence Details region is grayed out.

Site

A sequence number can be by installation site. One row is active in the Sequence Details region. This selection allows entry of Prefix, Suffix, Sequence Number From, Sequence Number To, Format Length, and Allow Manual override in the Sequence Details region.

Business Group, Operating Unit, Class, and Category

A sequence number can be generated for any of the above attributes. These attributes are selected from a LOV. Please refer to the Truth table for possible combinations of contract attributes for generating sequence numbers automatically.

User Function

A user defined function offers the flexibility to build in more logic for generating a sequence number. For example, you can have the contract number reflect the date and sequence when the contract becomes active (i.e., 03222001001). Please refer to the Truth table for possible combinations of contract attributes for generating a sequence number automatically.

Enforcing Number Generation

The Allow Manual check box indicates whether the contract number is a protected field or not. When this check box is not selected, the number will be generated automatically and this field is protected. If you select this check box the contract number will be automatically generated only when the contract number field is left blank.

Prefix and Suffix

Free form fields help provide 'classification' coding to a contract. Note that you can use these fields to add a separator or delimiter. These values are enter as characters only.

Sequence Numbers

You may choose to specify a starting and ending number for the numeric part of the contract number sequence. If the contract number exceeds the specified range, depending on the selection of the Allow Manual check box, you will be prompted to extend the range or enter a unique contract number.

Format Length

You can specify a formatting length for the numeric part of the contract number sequence. The total contract number length (concatenated suffix, number prefix) cannot exceed 120 characters.

Template

When a contract is created from a template, it generates a number automatically.

Copy Subcontract

When a contract is copied to a new contract, the attributes pertinent to the contract such as class and category will not change. The attributes such as Business Group and Operating Unit will be based on the responsibility using which the copy is being created and NOT those of the source contract. A new number will be generated accordingly.

3.15.3 Auto-Number Examples

The following table shows examples of the "First Number" based on the various combinations selected.

Business Group	Operating Unit	Class	Category	Prefix	Suffix	Sequence From	Sequence To	Length	First Number Generate
USA		Service		US-	-SRV	99	10000	5	US-00099-SRV
USA		Corporate		US-	-COR	55	50000	5	US-00055-COR
UK		Service		UK-	-SRV	1	20000	5	UK-00001-SRV
UK		Corporate		UK-	/COR	33	100000	6	UK-000033/COR
SP		License			/LIC	44		5	00044/LIC
MP			Network	MP-NW/		66	60000	5	MP-NW/00066
	OU					77		5	00077

3.15.4 Truth Table

Depending on which autonumbering attribute is selected, the Sequence Details fields are selectively enabled. For example, if only the Business Group is selected, then Operating Unit, Class, Category sequence details will be disabled. The following table ("Truth Table") shows the possible sequences:

Description	User Function	Site	Business Group	Operating Unit	Class	Category
User, no other attributes can be selected	Y	N	N	N	N	N
Site, no other attributes can be selected	N	Y	N	N	N	N
Business Group can be selected with Class	N	N	Y	N	Y	N
Business Group can be selected with Category	N	N	Y	N	N	Y
Business Group can be selected without Class or Category	N	N	Y	N	N	N

Description	User Function	Site	Business Group	Operating Unit	Class	Category
Operating Unit can be selected with Class	N	N	N	Y	Y	N
Operating Unit can be selected with Category	N	N	N	Y	N	Y
Operating Unit can be selected without Class or Category	N	N	N	Y	N	N
Class can be selected without Business Group/Operating Unit	N	N	N	N	Y	N
Category selected without Business Group/Operating Unit	N	N	N	N	N	Y

3.15.5 Setting Up Autonumbering

Prerequisites

The OKC: Document Sequence Name profile option should be enabled. This profile option is used to define a text literal, for example, OKC_DOC to be used for naming Sequences created.

The Sequential Numbering profile option should be set to “Always Used” or "Partially Used". By default, this profile is not set up. This profile is set at the application level. This means that Contracts for Service users may use sequential numbering while Contracts Core users can use manual numbering.

If Sequential Numbering Profile is set to "Always Used" or "Partially Used" and the Allow Manual flag is not selected in the AutoNumbering setup, Copy Contract API does not allow manual entry of Contract Number in the Copy form. It disables the Contract Number and Contract Number Modifier fields for the target contract.

Steps

1. From the Contracts Manager responsibility, choose Setup > Contract > Autonumbering.
2. Select the attribute type (Site, User Function, Business Group, and so on). If you select the User Function, choose a Process Name from the LOV.

3. Select the Sequence Details (depending on attribute selected see, the Truth Table for attribute combinations) from the LOV in the Sequence Details region.
4. Enter Sequence Numbers From/To (optional).
5. Enter the Format Length for the contract number (optional).
6. Enter Prefix/Suffix values (optional).
7. If you want to manually override the system-generated contract number, select the Allow Manual check box.
8. Save your work.
9. From the Contracts Manager responsibility, choose Launch Contracts > Open > Tools > New.
10. Select a Contract Category from the LOV.
11. Click Create.
12. In the Contract Authoring window verify that the contract number is correct based on your setup.
13. Repeat steps 9-12 to check contract number sequence.

Note:

If you try to change any of AutoNumbering attributes, for example from Class to Category attribute, a Decision window displays the following warning: "You are changing the current contract numbering setup. Existing current numbering rules will no longer be effective. Do you want to continue?". If you click Yes, anther window opens and asks: "Do you want to delete existing sequence numbering rules?". If you click Yes, the existing sequence numbering rules will be deleted and there is no way you can retrieve these sequence numbering rules. If you click No, the existing sequence numbering rules will be kept in the system. If later on you want to revert to the original attribute, such as Class attribute in the above example, the sequence numbering rules will appear when you change the attribute from Category to Class.

3.15.6 Setting Up User Function

A User Defined Function is a database procedure and must have the following parameters:

- x_contract_number OUT Varchar2
- x_return_status OUT Varchar2

These two parameters should not be defined in the Process Definition but instead should be included in the user-defined procedure.

Steps

1. A User Defined Function must be registered in the Process Definition form. From the Contracts Manager responsibility, choose Setup > Contract > Process Definition.
2. Enter Auto Numbering for Purpose, PLSQL for Type. Then enter the Package and Procedure you want to define.
3. Enter the following parameters:

Parameters	Description
Site	Implementation Site
Business Group	Business Group of the responsibility with which the contract is created
Operating Unit	Operating Unit of the responsibility with which the contract is created
Class	Class of contract
Category	Category of contract
Contract Currency	Currency code of contract
Contract Amount	Estimated amount for contract
Contract Party Information	A table of party roles and names to be passed as a record parameter

4. From the Contracts Manager responsibility, choose Setup > Contract > Autonumbering.
5. Check the User Function check box and select the user-defined process in the Process Name field.
6. If you want to allow manual override, select the Allow Manual check box.
7. Save your work.
8. From the Contracts Manager responsibility, choose Open > Contract Navigator > Tools > New.
9. Select a Contract Category from the LOV.

10. Click Create.
11. From the Contract Authoring window verify contract number is correct based on your setup.

Guidelines

- If you select the User Function check box in the Autonumbering form to generate contract number, you cannot select any sequence details in the Autonumbering form. The associated procedure would take care of autonumbering using any prefix, suffix, formatting etc.
- Parameter *x_contract_number* should return the generated unique contract number. Parameter *x_return_status* should be:
  - 'S' if the program succeeds and return a contract number
  - 'E' in case of error
  - 'U' for unexpected errors

3.16 Setting up XML Reports

This section includes the following topics:

3.16.1 XML Reporting Features Overview

Print Contract Data

Contracts Core has the option to run multiple reports on a single contract. These reports can be defined and customized to show contract data. Oracle Contracts Core supports contract previewing from the Contracts Authoring form. This

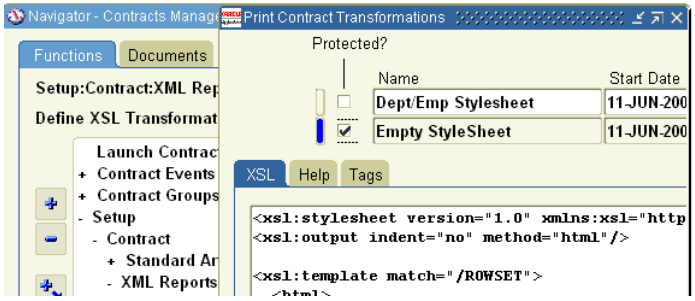
functionality allows for the formatting and output of contract attributes, such as contract header information, parties, lines, and articles.

There are two types of output: HTML and MS Word. HTML is used due to its simplicity and the ability for complete customization. If the PDF output is required, a PDF converter (such as Adobe Acrobat Distiller or Jaws PDF Creator) would allow you to convert and print content into a PDF file. If you need to view the report with MS Word, just save the report locally and open it in MS Word application.

There are two approaches for using Contract Reporting:

- Data Model + Layout approach is based on Oracle Contracts published data model (in terms of SQL and XML) and offered by the product layout as an XSL stylesheet that is open to modification.
- Keyword Definition lets you define keywords, such as Price List or Shipment information, and instructs what do with this information. These key words can be used in Contract Articles text enclosed in brackets, and specific Contract data will replace keywords when the Articles are printed.

Simplified flow for XML reporting with a relational database:



XML Reporting Terminology

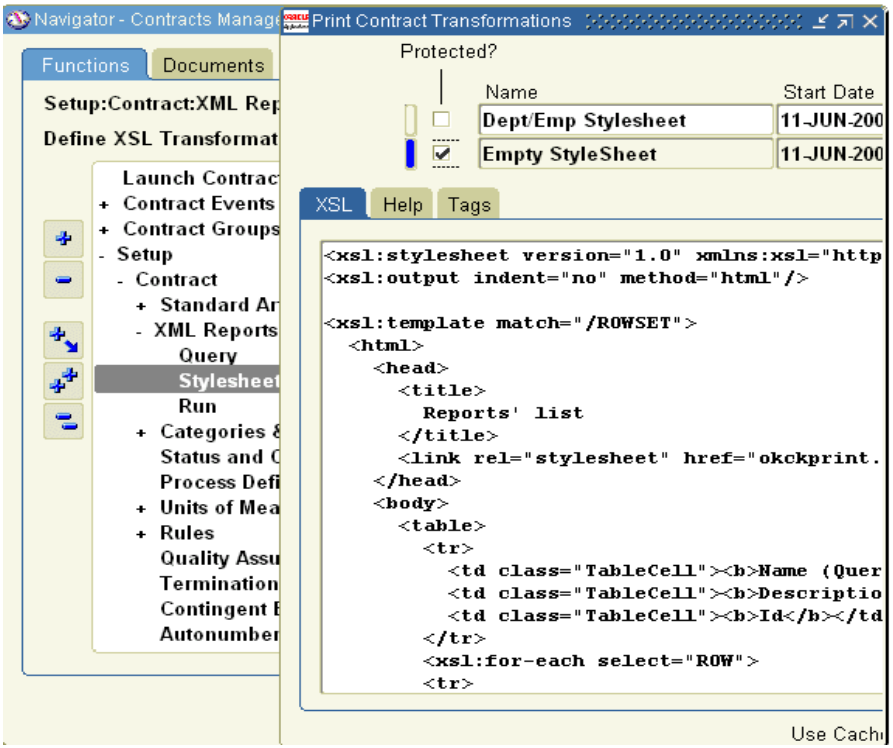
- JSP: Java Server Pages
- SQL: Query language for relational database
- XML: Extensible Markup Language
- XSU or SQL-XML Utility: The translator of relational object presentation into "XML" object
- XML Parser: The translator of one object presentation into another, i.e. provides something like a different view for the object

- XSL: The transformation language for XML that is used to describe rules for XMLParser how to change the object view
- XSLT: What XMLParser is doing: XML Transformation
- XDK: Oracle XML Kit, XSU and XMLParser are utilities from the Kit

Seeded SQL will automatically retrieve data from the contract and the linked XML takes the data, and formats it according to the XSL. The output (HTML) is defined in the XML report. Users can define layout using their own XSL directly in the setup forms provided by Contracts. This provides the flexibility to create a format according to each user's requirements.

The forms for setup include:

- Print Contract Data Model: Used to enter SQL query and contains SQL, Help, and Parameters tabs. From the Contracts Manager responsibility, navigate to Setup > Contract > XML Reports > Query.



- Print Contract Transformations: This contains the stylesheet language for XML report and contains XSL, Help, and Tags tabs. From the Contracts Manager responsibility, navigate to Setup > Contract > XML Reports > Stylesheet.
- Report List: You can use the Report List page to preview and print contracts from the Navigator window. You can also send the report to a FND user who, based on the FND user preference setup in Workflow Admin, is notified by E-mail or workflow notification. To open the Report List page, navigate to Setup> Contract> XML Reports> Run.

To run reports from the Contract Authoring form:

- Open the Report Type form. Navigate to: Contracts Authoring > Tools > HTML Preview or Tools > MS Word Preview.
- Select and run any report from the list of reports in the Report Type form.

Keyword Definitions (Tags)

Contract information consists of either Articles or information not contained in an article. Keywords are used to identify specific information in the contract that can be printed, such as contract number, parties, or lines within the text of an article.

The Sections tab in the Contracts Authoring form is used to organize the contract's Articles. Section labels and headings can be defined and articles can be organized in these sections. Sections can also be nested within other sections. The order of the contract information in Preview Contract should depend on the user defined sequencing of the articles.

Tags can be inserted in standard or non-standard articles by entering the tag directly into a non-standard article, or in the Varied Text field of a standard article. The forms you use for this are:

- Article Text form from the Articles tab is used when copying or referencing articles.
- Release Text form from the Standard Article Library setup page.

The values of these tags will then automatically be inserted into the article text upon generation of the report. For example, the tag representing the customer name is called <CUSTOMER_NAME/>; this can be embedded directly into an article. When the contract is previewed, instead of the tag, the customer name would actually appear within the article text.

Seeded Keywords (Tags)

A list of the seeded tags and their description is provided and can be viewed from the Print Contract Transformation window. From the Contracts Manager responsibility, navigate to Setup > Contract > XML Reports > Stylesheet. Select "Section/Articles/Data referenced by article tags" in the upper region of window. Click the Help tab. The list of seeded tags appears.

3.16.2 Set Up Contract Preview

The Contract Preview is a JSP page.

Steps

1. Check single sign-on setup. The single sign-on feature allows launching JSP pages from forms applications and allows moving from one application type into another, bypassing redundant sign-ons. Your application administrator should configure this option, but if you are having trouble when launching JSP reports from forms, check profile option values before asking your system administrator to check "iAS PL/SQL Gateway settings" or "APPS_WEB_AGENT status".

   To check profile option values:

   - Sign-on to Forms applications as the System Administrator
   - Select "System Administrator" responsibility
   - Navigate to Profiles > System.
   - Enter "%Agent%" profile options values, both at system level and the product level - Oracle Contracts Core.
   - Check these setups:
     - Applications Web Agent"value should look like http://[Hostname][:Port]/pls/[DAD] (for details check Metalink Note#146469.1)
     - Apps Servlet Agent" should look like http://[Hostname][:Port]/oa_servlets

   If these profile option values are empty that means some setup scripts have not been run and your system administrator should be contacted.

   You can override the second profile option (at the application level) to point to your Apache Port.

- Check if your JVM classpath contains 8.1.7 jdbc and xdk jars. For example:
  - /oracle/ias1022/jdbc/lib/classes12.zip (or jdbc12.zip that is apps copy)
  - /oracle/ias1022/lib/xmlparserv2.jar
  - /oracle/ias1022/rdbms/jlib/xsu12.jar

2. Set "Contract Printing" profile options:
   - Enter "OKC Report%" to check and set report profile options.

     If you preview and/or run reports from OKC "Contract Authoring form" you are to set "OKC: Contract Report" to any of the seeded reports and that will enable Tools/Print menu in Authoring form.
   - Check "OKC: Report prerun validation procedure" is set to the value as in the following row:
     - OKC_REPORT_PVT.prerun

     This procedure does not allow running reports when a contract has no lines, articles, or sections defined. You can bypass this validation by resetting this profile option value to null.

3.16.3 Testing Setup and Troubleshooting

Steps

1. Log-out and Log-in again in forms applications.
2. Select the Contracts Manager Responsibility.
3. Open or create a contract. Check or add lines and sections.
4. Save the contract.
5. From the Contracts Authoring form navigate to Tools > HTML Preview.
6. Select a demo report. For example, OKC: Articles.
7. The report should appear in your browser. If you get an error message here are possible failure scenarios:
   - "The requested URL /pls/…/OracleApps.LF was not found on this server...". This could indicate the “single signon” was not set properly. Contact your system administrator.

- If you get a message that some java class not found, contact your system administrator and send him a copy of the error message.
- If you get a message that something is wrong with OWA package or a buffer is too small, then:
  - Clean jcache and cookies: delete your PC files at "Program Files\Oracle\JInitiator 1.1....\jcache"
  - Open your browser and navigate to Tools > Internet Options.
  - Click [Delete Cookies] or [Delete Files] buttons and next button [Settings...] as well. The last click will open another window.
  - Click [View Files...], select all files and delete them.
  - Log-on to Contracts again and check if Printing works.

3.16.4 Registering a Report

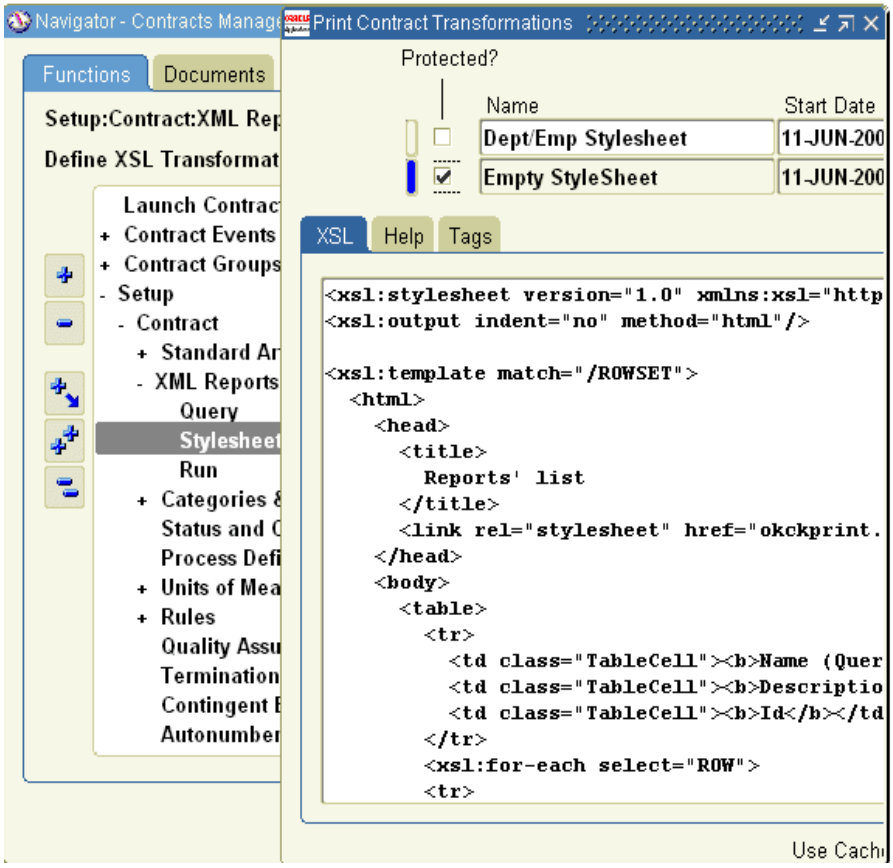
This procedure guides you through the Contracts Core forms that are used to define and test your report. The two Contracts Core setup forms and Contracts Authoring form "Print" can be considered as simplified sql-xml-html tool (wrapper on top of Oracle XDK) and is a good starting point in getting to your own report definitions.

To master SQL-HTML printing, it is recommended to use XSQL and ORAXSL command line utilities.

3.16.4.1 Query Registration

Prerequisites

- Experience with SQL
- Knowledge of XML/XSL



Steps

1. From the Contracts Manager responsibility, navigate to Setup > Contract > XML Reports > Query > SQL.
2. Enter the following query:

```
select
  dept.*,
  cursor (
    select emp.*
    from emp
    where emp.deptno = dept.deptno
    order by 1) employees
 from dept
```

```
order by 1
```

3. Enter "Dept/Emp test" for query name. This field is located in the top region of the form.

   Any modifications to the SQL done by Oracle Users (during setup) will work at the XSL level mostly, but it is good to understand what data to layout. It is recommended to test every new query in sql*plus first as user APPS. If you have “visibility problems” then contact your System Administrator.

4. Navigate to Setup > Contract > XML Reports > Stylesheet.
5. Select an Empty Stylesheet.
6. Enter "Dept/Emp test" to the Apply to Datamodel.

3.16.4.2 Running the Query

Steps

1. In the Contracts Manager responsibility, navigate to Launch Contracts > Open.
2. Select and open any contract.
3. Navigate to Tools > HTML Preview.
4. Select "Dept/Emp test".
5. Click OK. The browser window appears.
6. Navigate to View Source.
7. Save the file with.xml extension.
8. Double click file to view datagram produced by SQL-XML utility. The following is representative of the datagram:

```
<?xml version="1.0" encoding ="UTF-8"?>
- <ROWSET>
 - <ROW num="1">
    <DEPTNO>10</DEPTNO>
    <DNAME>ACCOUNTING</DNAME>
    <LOC>NEW YORK</LOC>
  - <EMPLOYEES>
     <EMPLOYEES_ROW num="1">
```

```
        <EMPNO>7782</EMPNO>
         <ENAME>CLARK</ENAME>
         <JOB>MANAGER</MANAGER>>
         <MGR>7839</MGR>
         <HIREDATE>6/9/1981 0:0:0</HIREDATE>
         <SAL>3424</SAL>
         <DEPTNO>10</DEPTNO>
      </EMPLOYEES_ROW>
    + <EMPLOYEES_ROW num="2">
    + <EMPLOYEES_ROW num="3">
    </EMPLOYEES>
  </ROW>
+ <ROW num="2">
+ <ROW num="3">
 <ROW num="4">
 <ROWSET>
```

3.16.4.3 Assigning the Stylesheet and Producing the HTML

Using this procedure, you can copy this sample stylesheet into the XSL tab of the Print Contract Transformations form.

Steps

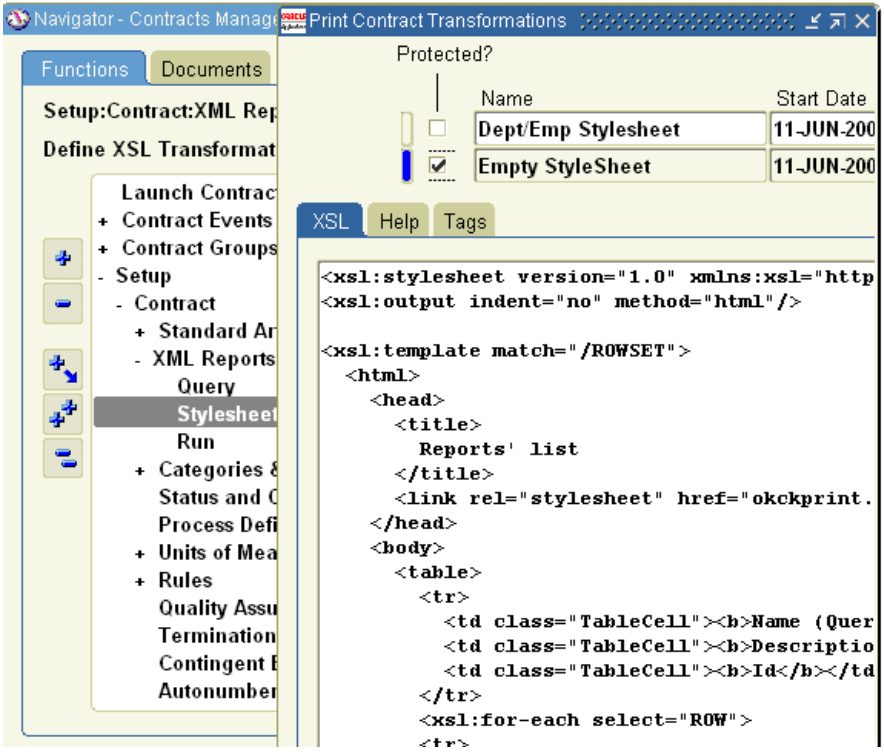
1. Copy this stylesheet text.

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output indent="no" method="html"/>

<!--This template produces body for the html document -->
      <xsl:template match="/ROWSET">
        <html>
          <head>
            <title>Dept/Emp test</title>
          </head>
          <body>
```

```
            <xsl:apply-templates/>
          </body>
        </html>
      </xsl:template>

      <!--This template outputs dept as record and its employee list -->
      <xsl:template match="//ROW">
        <table>
          <tr>
            <td>
              <h3><xsl:value-of select="DEPTNO"/>. <xsl:value-of
select="DNAME"/></h3>
            </td>
          </tr>
          <xsl:apply-templates select="EMPLOYEES"/>
        </table>
      </xsl:template>

      <!-- This template produces header for the employees list -->
      <xsl:template match="//EMPLOYEES">
        <tr>
          <td>
            <table>
              <tr>
                <th>#</th>
                <th>Employee</th>
                <th>Salary</th>
              </tr>
              <xsl:apply-templates/>
            </table>
          </td>
        </tr>
      </xsl:template>

      <!--This prints out employee information -->
      <xsl:template match="//EMPLOYEES_ROW">
        <tr>
          <td><xsl:value-of select="EMPNO"/></td>
          <td><xsl:value-of select="ENAME"/></td>
          <td><xsl:value-of select="SAL"/></td>
        </tr>
      </xsl:template>

      <!-- do nothing in other cases -->
      <xsl:template match="node()|@*">
```

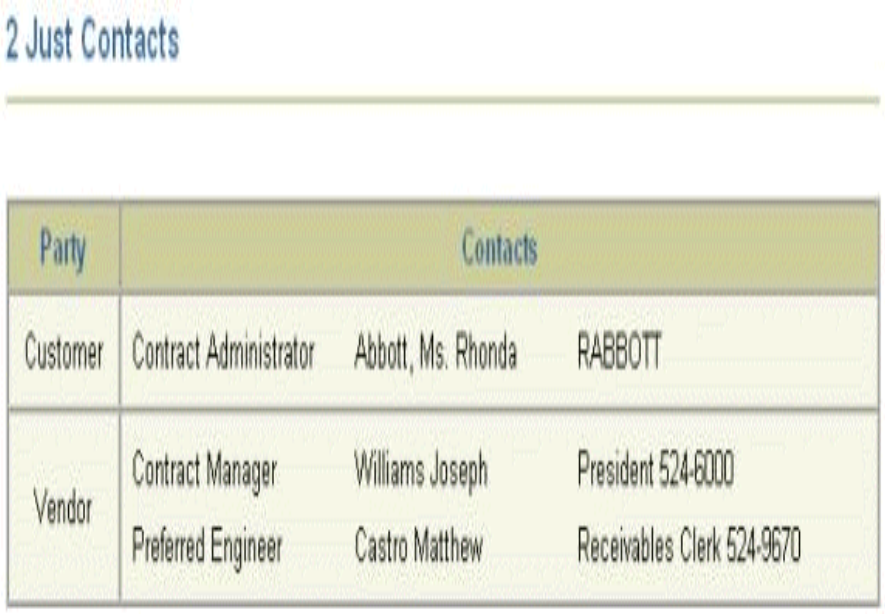
```
</xsl:template>

</xsl:stylesheet>
```

2. From the Contracts Manager responsibility, navigate to Setup > Contract > XML Reports > Stylesheet > XSL
3. Paste the stylesheet into the Print Contract Transformations XSL tab.
4. Assign this stylesheet for "Dept/Emp test".



5. Save your work.
6. From the Contracts Manager responsibility, navigate to Launch Contracts > Open.
7. Select and open any contract.

8. Navigate to Actions > Print.
9. Select "Dept/Emp test".
10. View the HTML page.



2 Just Contacts

Party	Contacts		
Customer	Contract Administrator	Abbott, Ms. Rhonda	RABBOTT
Vendor	Contract Manager	Williams Joseph	President 524-6000
	Preferred Engineer	Castro Matthew	Receivables Clerk 524-9670

3.16.5 Creating a Contracts Report

3.16.5.1 Publishing the Data Model

In the queries for XML Preview shipped by Oracle Contracts, you will see that "Contract Id" is the value returned by "okc_tree_index.get_root_id" function. Same way any Contracts product should reference it within registered queries. The API takes care of setting this package variable using "child" request parameter value. Sample query:

```
select * from okc_k_headers_v
```

…

where id= okc_tree_index.get_root_id

For this example, information will be printed related to the Contract Header and Contract Lines. If you are interested in "Lines Hierarchy" then you can use Contracts Core provided treewalk view:

okc_lines_index_v(ID,LINE_INDEX,LINE_ORDER)

Where ID stands for the Line ID, LINE_INDEX stands for a hierarchy Index to display, and LINE_ORDER stands for same index in format convenient for ordering.

Here is a sample "Contract data model":

SQL-XML Utility driven by this SQL statement will produce a document of the following structure:

```
SELECT
  K.CONTRACT_NUMBER||'-'||K.CONTRACT_NUMBER_MODIFIER K_NUMBER,
  KV.major_version||'.'||KV.minor_version K_VERSION,
  K.SHORT_DESCRIPTION,
  STS.meaning STATUS,
  to_char(K.START_DATE) START_DATE,
  to_char(K.END_DATE) END_DATE,
  cursor(
    select
      TR.line_index,
      LCR.SYMBOL||to_char(L.PRICE_UNIT,
        fnd_currency.GET_FORMAT_MASK(L.CURRENCY_CODE,40)) UNIT_PRICE,
      ITM.NUMBER_OF_ITEMS,
      LCR.SYMBOL||to_char(L.price_unit * ITM.number_of_items,
        fnd_currency.GET_FORMAT_MASK(K.CURRENCY_CODE,40))EXTENDED_PRICE,
      LS.NAME LINE,
      decode(LS.LTY_CODE, 'FREE_FORM', L.NAME,
        okc_rule_pub.get_object_val(
          ITM.JTOT_OBJECT1_CODE, ITM.OBJECT1_ID1, ITM.OBJECT1_ID2)) ITEM
     from
       okc_lines_index_v TR,
       okc_k_lines_v L,
       okc_line_styles_v LS,
       okc_k_items ITM,
       fnd_currencies LCR
      where L.id = TR.id
        and LS.ID = L.LSE_ID
        and ITM.CLE_ID (+) = L.ID
```

```
        and LCR.currency_code (+) = L.currency_code
      order by TR.line_order
    ) LINES
FROM
  OKC_K_HEADERS_V K,
  OKC_K_VERS_NUMBERS_V KV,
  OKC_STATUSES_V STS
WHERE
  K.id = okc_tree_index.get_root_id
  and KV.chr_id = K.id
  and STS.code = K.sts_code
```

The SQL-XML Utility driven by this SQL statement will produce a document of the following structure:

```
<ROWSET>
  <ROW>
     <K_NUMBER/>
     <K_VERSION/>
     <SHORT_DESCRIPTION/>
     <STATUS/>
     <START_DATE/>
     <END_DATE/>
     <LINES>
        <LINES_ROW>
           <LINE_INDEX/>
           <UNIT_PRICE/>
           < NUMBER_OF_ITEMS />
           <EXTENDED_PRICE/>
           <LINE/>
           <ITEM/>
         </LINES_ROW>
         ...
     </LINES>
   </ROW>
</ROWSET>
```

This structure can be pasted in the Help tab for this query (in the Data Model registration form). As for more complete query samples, review "OKC: Contracts sample".

Contracts Products will publish their Data Models. To optimize performance, do not try to put all your tables/views in single SQL statement; you can publish several SQL statements that are Task/Report oriented.

3.16.5.2 Suggested XML Contract Layout

Using the previous data model this sample style sheet is used as a single template that provides an output showing the contract header first and then the contract lines.

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output indent="no" method="html"/>

<xsl:template match="/ROWSET/ROW">
  <html>

    <head>
      <title>
        <xsl:value-of select="K_NUMBER"/> v.<xsl:value-of select="K_
VERSION"/>
      </title>
      <link rel="stylesheet" href="okckprint.css" type="text/css"/>
    </head>

    <body>

      <table>
        <tr>
          <td height="68" width="325"                  background="../OA_
MEDIA/okcCrmCollage.gif">
          </td>
          <td valign="middle" class="header">
            Sales Contract
          </td>
        </tr>
      </table>
      <HR size="8"/>

      <table width="100%" border="0" cellspacing="0" cellpadding="10">

        <tr>
          <td>
            <table width="100%" border="0" cellspacing="0" cellpadding="10">
              <tr>
                <td class="subheader1">
                  Contract Information<HR size="2"/>
                </td>
               </tr>
                 <tr>
                   <td>
```

```
                    <table width="100%" border="0" cellspacing="0"
cellpadding="5">
                       <tr>
                         <td width="20%" align="right">Contract: </td>
                         <td width="*" align="left">
                           <b><xsl:value-of select="K_NUMBER"/>
           v.<xsl:value-of select="K_VERSION"/></b>
                         </td>
                       </tr>
                       <tr>
                         <td width="20%" align="right">Short Description:
</td>
                         <td width="*" align="left">
         <b><xsl:value-of select="SHORT_DESCRIPTION"/></b>
       </td>
                       </tr>
                     </table>
                   </td>
                 </tr>
               </table>
             </td>
           </tr>

           <tr>
             <td>
               <table width="100%" border="0" cellspacing="0"
cellpadding="10">

                 <tr>
                   <td class="subheader1">
                     Products<HR size="2"/>
                   </td>
                 </tr>
                 <tr>
                   <td>
                     <table width="100%" border="1" cellspacing="0"
cellpadding="4">
                       <tr>
                         <td width="15%" class="TableCell"><b>Index</b></td>
                         <td width="40%" class="TableCell"><b>Item</b></td>
                         <td width="15%" align="right"
class="TableCell"><b>Unit Price</b></td>
                         <td width="10%" align="right"
class="TableCell"><b>Quantity</b></td>
                         <td width="20%" align="right"
```

```
class="TableCell"><b>Extended Price</b></td>
                    </tr>
                   <xsl:for-each select="LINES/LINES_ROW">
                    <tr>
                      <td width="15%">
                        <xsl:value-of select="LINE_INDEX"/>
                      </td>
                      <td width="40%">
                         <xsl:value-of select="ITEM"/>
                       </td>
                       <td width="15%" align="right">
                              <xsl:value-of select="UNIT_PRICE"/>
                            </td>
                            <td width="10%" align="right">
                              <xsl:value-of select="NUMBER_OF_ITEMS"/>
                            </td>
                            <td width="20%" align="right">
                              <xsl:value-of select="EXTENDED_PRICE"/>
                            </td>
                          </tr>
                        </xsl:for-each>
                        </table>
                      </td>
                    </tr>

                  </table>
                </td>
              </tr>

            </table>
          </body>
        </html>
      </xsl:template>

      </xsl:stylesheet>
```

You can register this stylesheet and assign it to the previously registered Data Model. Open any contract and preview it.

3.16.6 Using Keyword Definitions (Tags)

This section reviews how to create a report based on Articles content and it will print expected data from the Contract in-place based on keywords.

Here is how such Articles look:

<CUSTOMER/> and <VENDOR/> have signed this agreement …

… following products

<PRODUCTS_LIST/>

…

Signatures: on behalf of <CUSTOMER/> on behalf of <VENDOR/>

…

Keywords (“tokens” or “tags”) are enclosed with brackets and a closing slash based on XML standards. The keywords will be replaced with actual Customer and Vendor names and an actual List of Products when the contract is previewed. The content of the tag could be simple or complex (and is based on the mock-up), and could be implemented in a short time. With this approach, users eventually will compile their own Articles and keywords Library that could be extended, updated and maintained. In that case the stylesheet resembles a dictionary that explains what data will appear and how it will be formatted in place of these keywords. When authoring articles, these keywords can be inserted directly into the article text (the author should be able to get a list of available keywords from the help text).

To achieve this:

- In published SQL articles columns should have alias “TEXT”: this alias will let XmlParser into articles text.
- Stylesheet should decode in-TEXT tags. For example: Articles author requested keyword CONTACTS_LIST. The engineer should write html implementation for CONTACTS_LIST (found in TEXT):

```
<xsl:template match="//TEXT/CONTACTS_LIST">
    <table width="100%" border="1" cellspacing="0" cellpadding="4">
      <tr>
        <td width="70" class="TableHeader" align="center">Party</td>
         <td width="*"  class="TableHeader" align="center">Contacts</td>
       </tr>
     <xsl:for-each select =                      "/ROWSET/ROW/CONTACTS/CONTACTS_
ROW[not(preceding::CPL_ID=CPL_ID)]">
      <tr class="TableCell">
        <td width="70" align="center" valign="middle">
          <xsl:value-of select="PARTY_ROLE"/>
        </td>
        <td width="*">
        <table width="100%" border="0" cellpadding="2" cellspacing="2">
          <xsl:for-each select =            "/ROWSET/ROW/CONTACTS/CONTACTS_
ROW[CPL_ID=current()/CPL_ID]">
```

```
            <tr>
              <td width="30%"><xsl:value-of select="CONTACT_ROLE"/></td>
              <td width="30%"><xsl:value-of select="NAME"/></td>
              <td width="*"><xsl:value-of select="DESCRIPTION"/></td>
             </tr>
           </xsl:for-each>
           </table>
         </td>
       </tr>
     </xsl:for-each>
     </table>
</xsl:template>
```

Here is the resulting report (if article text is "… < CONTACTS_LIST />…"):

2 Just Contacts

Party	Contacts		
Customer	Contract Administrator	Abbott, Ms. Rhonda	RABBOTT
Vendor	Contract Manager	Williams Joseph	President 524-6000
	Preferred Engineer	Castro Matthew	Receivables Clerk 524-9670

The "//TEXT/CONTACTS_LIST" template shows one more XSL feature:

XSL by itself is capable of breaking data if by some reason this work has not been done at the SQL level. For a very comprehensive stylesheet sample, review "Sections/Articles/Data referenced by articles tags" provided by OKC. The engineer should maintain an Articles tags list supported by the stylesheet and keep Help text for Articles writer in sync. From the Contracts Manager responsibility, navigate to Setup > Contract > XML Reports > Stylesheet. Click on the Help tab. For instance, here is such a help text for the mentioned Contracts Core stylesheet:

```
Token                    Meaning
-----------------------  ------------------------------------------
      <ATTACHMENTS/>           Contract attachments
      <CONTRACT_NUMBER/>       Contract number
      <CONTRACT_MODIFIER/>     Contract modifier
      <CONTRACT_VERSION/>      Contract version
      <SHORT_DESCRIPTION/>     Short Description
      <AMOUNT/>                Estimated amount with currency code
      <CONTRACT_CURRENCY/>     Currency code
      <STATUS/>                Status
      <START_DATE/>            Start date
```

```
<END_DATE/>              End date
<DATE_SIGNED/>           Date signed
<DATE_PRINTED/>          Date printed
<CUSTOMER_NAME/>         Customer
<CUST_ADMIN_NAME/>       Administrator (Customer side)
<CUST_ADMIN_POINT/>      Description/contact point
<CUST_TECH_NAME/>        Technical contact (Customer side)
<CUST_TECH_POINT/>       Description/contact point
<SUPPLIER_NAME/>         Vendor
<SUPPL_ADMIN_NAME/>      Administrator (Vendor side)
<SUPPL_ADMIN_POINT/>     Description/contact point
<BILL_TO_ADDRESS/>       Customer address
<SHIP_TO_ADDRESS/>       Customer address
<PAY_TO_ADDRESS/>        Producer address
<CONTACTS_TABLE/>        List of Contacts
<RULES_TABLE/>           "Other details" for the Contract
<LINES_TABLE/>           Products list
<PARTIES_DETAILS/>       Parties/Contacts info
<LINES_DETAILS/>         Lines info
<SIGNATURES/>            Signatures block
```

Sign 'less' is reserved for token start, if you need it for other purposes use & lt;. If you need ampersand as 'and' use &.

3.16.7 Advanced Topics

3.16.7.1 Working with Nested Queries

Your Contract Data Model can be nested, for example, you might have a contract structure like this:

Header

Parties

Party Contacts

Lines

Rules for Lines

If you use XDK 8.1.7, you can experience SQL-XML transformation exceptions when your contract has no record entered at some intermediate level: i.e. XSU will crash if there is no Party or no Line defined for this sample data model. To avoid this situation do one of the following:

- Set "OKC: Report prerun validation procedure" profile option value to the procedure that will validate the contract before report launching. For example Contracts Core ships OKC_REPORT_PVT.prerun procedure that checks if the contract has lines and sections. Your procedure should have the same signature as “OKC_REPORT_PVT.prerun” and be oriented to your data model.
- Join nested queries in outer joins, like Parties & Contacts(+). And you can restore nested structure with XSL as was shown in the previous section. Or you can (this way you improve query performance as well) query Parties and Contacts in separate cursors and restore relations between them using XSL.

3.16.7.2 Transformation Exceptions

If you get an "Unhandled exception (XSLT failed), please verify stylesheet" error, check the following:

- The stylesheet for any typographical errors
- If in the stylesheet you declared other namespaces mapped to your java classes, check if these classes are in JVM classpath (even if this class is in apps.zip, apps.zip should be listed in JVM CLASSPATH)
- If you used articles as stylesheet extensions (as described before with in-Articles tags) then, check if such tags are closed with a slash, i.e.: <PRICE_LIST/>

You can use tools to debug your stylesheets such as Oracle XSLT utility “oraxsl”.

3.16.7.3 Report Cache

The “Use cache?” flag appears in both report definition screens. If this flag is set, then the SQL-XML or XML-HTML transformation result will be saved in the database and on subsequent printing requests (if Contract has not been changed in between) transformations will be bypassed and cached information will be used instead. Do not set this flag if your report is date dependent. For some tasks this flag should be reset, for instance if your report is run-date-dependent. One more reason for not setting this flag is discussed in *Securing Online Links*.

Saved reports could be used in post processing step for sending integration or E-mail messages: your product can launch some workflow just by mapping “OKC: After Report procedure” profile option value on some PL/SQL procedure that does required work.

3.16.7.4 Session Specific Information

For your report you might need NLS or some other session specific information. There are two ways to deliver this information: with the SQL query or with stylesheet parameters. This represents a sample of "OKC: Contracts sample" query that contains some session information:

```
Select …
 fnd_profile.value('APPS_SERVLET_AGENT') HTML_HEAD_BASE,
 replace(replace(replace(replace(replace(
  fnd_currency.GET_FORMAT_MASK(K.CURRENCY_CODE,40),
  'PR',''),'FM',''),'9','#'),
  'D',substr(NLS.value,1,1)),
  'G',substr(NLS.value,2,1)) CURRENCY_MASK,...
```

HTML_HEAD_BASE could be used to provide path to css and image files (useful for offline copies viewing):

```
<head>
   <base href="{/ROWSET/ROW/HTML_HEAD_BASE}"/> ...
```

It is up to you to delete it if you do not need it in offline report copies. Or it is up to the user to have a collection of intranet images and styles, then the user can put intranet <base href> values directly in the stylesheet.

As for currency mask, it is used for summaries formatting at the stylesheet level (base table data better to format in SQL query itself):

```
<xsl:value-of select="format-number(sum (//LINES_ROW/EXTENDED_
PRICE),/ROWSET/ROW/CURRENCY_MASK)"/>
```

3.16.7.5 Securing Online Links

Some URLs should be wrapped with framework session information. It is required with Oracle CRM standards to let these URLs get into the database when cookies are disabled. Because secure links will not work when called from saved report copies, it is not recommended to cache HTML reports if it contains such links (still you can cache report at SQL-XML level, and that is recommended). So do not set cache flag for XSLT in this case. To secure URLs, add to them FWK_ADDON parameter that you are to define at the top of the stylesheet:

```
<--! Add this parameter to your stylesheet -->
```

```
<xsl:param name = "FWK_ADDON"></xsl:param>
...
<--! and add FWK_ADDON to the secure links with ? or & -->
{URL}&amp;{$FWK_ADDON}
...
```

To protect cache from taking session dependent information $FWK_ADDON will be empty if XSLT Cache is on (even when cookies are disabled).

3.16.7.6 Printing with HTML Styles

Styles useful for printing with IE are listed here (all of them tested with IE 5.5), if you work with different browser check whether it is CSS2 compliant.

To replicate table header/footer across pages use “display: table-header/footer-group” styles:

```
...
<style>
thead { display:table-header-group }
tfoot { display:table-footer-group }
</style>
...
<table>
  <thead><tr>
    <td>...</td>
    <td>...</td>
  </tr></thead>
  <tr>
    <td>...</td>
    <td>...</td>
  </tr>
...
  <tfoot><tr>
    <td>...</td>
    <td>...</td>
  </tr></tfoot>
</table>
...
```

To show/hide window title, date, page#/total pages at header/footer use browser settings from "File"/"Page setup" menu.

To enforce page break use “page-break-before/after” styles:

```
<p style="page-break-before:always; page-break-after:always">
  ...
</p>
```

To avoid page breaks within some text enclose it in thead tags:

```
...<style>thead { display:table-header-group }</style>...
<table>
  <thead><tr><td>
    ... this piece is unbreakable ...
   </td></tr></thead>
</table>
```

To preserve white spaces (i.e. spaces, tabulations, line breaks) when Articles text is wrapped use "word-wrap: break-word" style combined with absolute column widths:

```
<xsl:template match="ARTICLES_ROW">
  <table border="0" cellspacing="0" cellpadding="10">
  <col width="650" />
    <tr>
      <td class="subheader1"><xsl:value-of
select="LABEL"/> <xsl:value-of select="NAME"/>
      <HR size="2"/></td>
    </tr>
    <tr>
      <td><xsl:apply-templates select="TEXT"/></td>
    </tr>
  </table>
</xsl:template>

<xsl:template match="//TEXT">
  <pre style="word-wrap:
break-word"><xsl:apply-templates/></pre>
</xsl:template>
```

3.16.7.7 Java Function Calls in XSLT

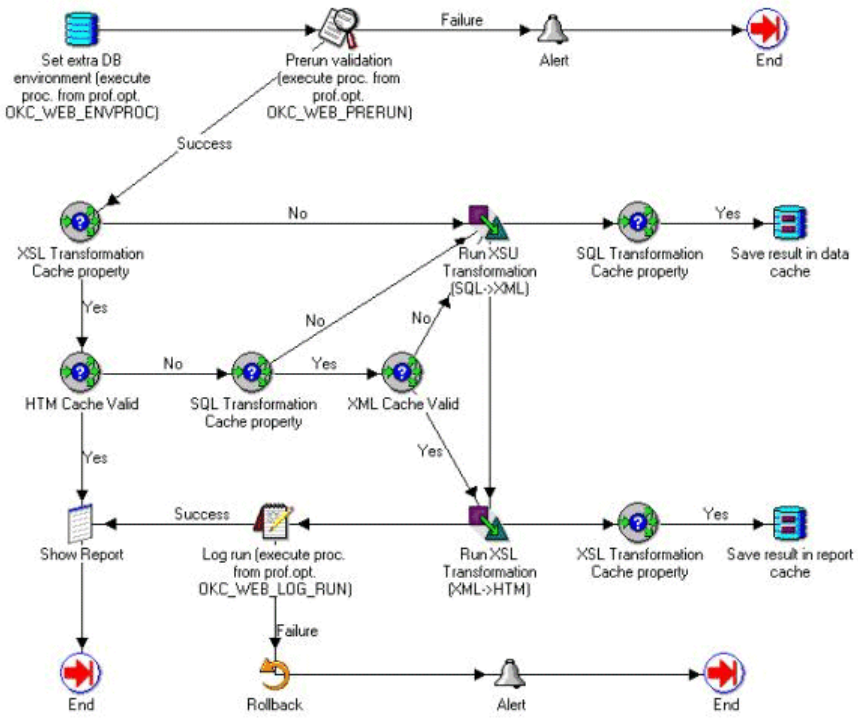
If you need a function that is not in XSLT specification, you can write program in java and call it within stylesheet. For this, you should map a new namespace to your java class and make sure that this java class is in your JVM classpath. Here is sample call for static doDiff method from oracle.apps.okc.common.shared.text.StandardHtmlDiff class (complete text is in “Highlight Articles Variations” stylesheet):

```
<xsl:stylesheet version="1.0" exclude-result-prefixes="Diff"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:Diff =
"http://www.oracle.com/XSL/Transform/java/oracle.apps.okc.common.shared.text.Sta
ndardHtmlDiff">
...
```

```
 <xsl:value-of select="Diff:doDiff(STD_TEXT,NON_STD_TEXT)"
disable-output-escaping="yes"/>
...
</xsl:stylesheet>
```

3.16.8 Execution

This diagram shows what happens "behind the scene" when a request is made for Contract Preview:



Few profile options (by default they are not set) can be used to embrace report run with proper environment, pre-run and post-run procedures. They are supposed to be used by Oracle Contracts products at the product or responsibilities levels.

3.17 Setting up Discoverer Reports

Discoverer Reports is a reporting feature which allows you to search for desired contract data and export this data to a customizable report such as HTML, MS Excel spreadsheet, or other type of output.

This topic group contains the following procedures:

Setting up

Testing Setup

Troubleshooting Setup

3.17.1 Setting up

The following procedures are part of setup:

- Setting Profile Options
- Creating a EUL
- Assigning Privileges
- FTP Files
- Importing Files
- Validate Business Area
- Validate Workbook

References

Discoverer 4i with Oracle Applications 11i (Oracle*MetaLink*)

Oracle OKC Post Upgrade Steps for Discoverer (Oracle*MetaLink*)

Discoverer 4i End User Layer Maintenance to Contracts 11i (Oracle*MetaLink*)

Prerequisites

- You must install or upgrade Oracle Applications 11*i* to Discoverer 4*i*. See *Discoverer 4i with Oracle Applications 11i* for more information.
- Install patch # 1755161 for database side objects for Oracle Contracts Core Discoverer reporting.
- Ensure your site has the KMKS Korn Shell tool. This enables unix operations on Windows NT.

Steps

3.17.1.1 Setting Profile Options

There are two Contracts Core profile options for Discoverer Reports:

Profile	Description	Access Level
OKC: Discoverer Reports Enabled	To enable site specific Discoverer reports feature.	Site
OKC: Contracts Expired Past Days	To allow you to set the value of the N number of days past a contract which has expired. For example, Contracts expired for the past 60 days, the profile option will contain the value 60. This value is utilized in the search criteria.	User

1. From the System Administrator responsibility, navigate to Profile > System. The Find System Profile Values window appears.
2. Perform a query for profile.
3. Enter values.
4. Close the form and save changes.

3.17.1.2 Creating a EUL

You can skip this procedure if a EUL4_US already exists. If not then follow the steps outlined in the procedure *Creating an Applications Mode URL, Oracle Discoverer Administration Edition Administration Guide.*

3.17.1.3 Assigning Privileges

Use this procedure is to ensure successful EUL4 user creation and also assign privileges to Oracle Contracts Core responsibility and Oracle Contracts Core users.

1. To connect to Discoverer, see, *Connect as an Applications User, Oracle Discoverer Administration Edition Administration Guide.* If your login is successful, you will see the Load Wizard window.
2. Click Cancel. You will not be creating any business areas manually. The import process will create the business area along with two folders and two workbooks.

3. From the Tools menu select Options. The Options window opens.
4. From the Connection tab, select "Connect to both standard and application EULs".
5. Enter "applsyspub/pub" for Gateway User ID.
6. Enter "apps" for Foundation Name.
7. Click OK.
8. From the Tools menu select the Privileges option. The Privileges window opens.
9. From the Privileges tab, select the Responsibility check box.
10. Select a responsibility from the LOV. For example "Contracts Manager".
11. Select all check boxes within Administration.
12. Click OK.
13. Repeat steps 9 through 12 for additional responsibilities.
14. Select the User check box.
15. Select a User from the LOV. For example "OKCTEST".
16. Select all check boxes within Administration.
17. Click OK.
18. Repeat steps 14 through 17 for each additional user.
19. To test the access for the users that you just added, connect as one of the users. If you see Load Wizard window the connection was successful.
20. Click Cancel.

3.17.1.4 Importing Files

Using this procedure you are importing eex files via the Korn Shell script for OKC specific files. You will create a new script file similar to the one shown below in the same root folder. This file will act as a parameter file to execute the Oracle Contracts Core specific details to be passed on to the Oracle specific shell script "adupdeul.sh. The okcupdeul.sh log file contains the import details and the final return status.

This script and other related scripts are available through the Discoverer 4i install.

1. Navigate to the directory folder on your local drive where you placed the eex and sh files.

1. Open the "okcupdeul" (Kornshell Script File). The MKS Korn Shell window appears.
2. Enter command "cat okcupdeul.sh" at the prompt. A script appears that looks similar to the following:

```
sh adupdeul.sh connect=okctest/welcome@kxbuildi resp="Contracts Manager,
Vision Enterprises" gwyuid=APPLSYSPUB/PUB fndnam=APPS secgroup="Standard"
topdir=d:/bblock/import/okc language=US eulprefix=EUL4 eultype=OLTP
mode=COMPLETE exedir=D:/orant/discvr4 logfile=okcupdeul.log
```

3. Replace as needed with your local setup values: corresponding database, user, password, responsibility, topdir, exedir, and logfile.
4. Enter "sh okcupdeul.sh" at the prompt. You should receive "You are running adupdeul, version 115.8" message. The command line arguments should display as part of the log. The last lines for log should show the import status.

3.17.1.5 Validate Business Area

Use this procedure is to ensure the newly created Business Area and accompanying folders exist. Enter the same User Name selected for the import.

1. From the Oracle Discoverer module, navigate to the Connect to Oracle Discoverer Administration Edition window.
2. Enter the User Name.
3. Enter the Password.
4. Enter the database name for Connect.
5. Click Connect. The Load Wizard window appears.
6. Select "Open an existing business area".
7. Select "Oracle Contracts Core".
8. Click Finish. The Workarea window opens.
9. From the Data tab, expand folder nodes to display all fields.
10. Navigate to the Main menu and select View > Validate Folders. This is to ensure there are no errors for database access and privileges.
11. Close the Workarea window.

3.17.1.6 Validate Workbook

Use this procedure is to ensure the newly created workbooks exist. Enter the same User Name selected for the import.

1. From the Oracle Discoverer module, navigate to the Connect to Oracle Discoverer window.
2. Enter the User Name.
3. Enter the Password.
4. Enter the database name for Connect.
5. Click Connect. The Workbook Wizard window opens.
6. Choose "Open an existing workbook".
7. Choose "Database" from where you want to open workbook. The Open Workbook from Database window opens. There are two workbooks appearing in the "Choose a workbook to open":
   - Contracts Listing
   - Expired Contracts

3.17.2 Testing Setup

1. From the Open Workbook from Database window, select "Contract Listing" for a workbook to open.
2. Click Open. The Workbook in Other Database Account window appears.
3. Select "Open the workbook in the current database account".
4. Click OK. An Oracle Discoverer query window appears asking you "Do you want to run the query for the sheet "Sheet 1"?".
5. Click Yes. The Parameter Wizard window appears.
6. Click Finish. At this point there is no data to report, because the report is not run from the Contracts Launchpad yet. The Oracle Discoverer window appears.
7. Click OK. The Workbook window appears. This window indicates the workbook has been imported successfully.
8. Close the window.

Note: At this point you can share the workbooks with multiple users, who have been privileged. From Discoverer, Navigate to File > Manage Folders > Sharing. The Sharing window appears.

The remaining steps provide instruction for testing the Expired Contracts report.

9. From the Open Workbook from Database window, select the Expired Contracts.
10. Click Open. The Workbook in Other Database Account window opens.
11. Select the Open the Workbook in the Current dDtabase Account option.
12. Click OK. An Oracle Discoverer Query window opens and displays the following question: "Do you want to run the query for the sheet "Sheet 1"?"
13. Click Yes. The Parameter Wizard window opens.
14. Click Finish. At this point there is no data to report, because the report is not run from the Contracts Launchpad yet. The Oracle Discoverer window appears.
15. Click OK. The Workbook window appears. This window indicates the workbook has been imported successfully.
16. Close the window.

Note: At this point you can share the workbooks with multiple users, who have been privileged. From Discoverer, Navigate to File > Manage Folders > Sharing. The Sharing window appears.

Now you should be able to invoke the reports from the Contracts Launchpad. For this procedure see, *Oracle Contracts Core Users Guide.*

3.17.3 Troubleshooting Setup

The following represent possible error messages along with the possible solution:

1. Unable to connect to sysadmin@your_database_name. Error occurred while fetching the list of Oracle Applications responsibilities. Database error: ORA-00942: table or view does not exist.

   Solution: Ensure that the EUL Schema is created for "Oracle Applications use only" by selecting the check box in the Create EUL wizard i.e. while creating a

EUL4_US user with the Discoverer Administration Edition. Initial login should be "apps user".

2. This user does not have permissions to access an EUL. Do you wish to create one?

   Solution: Ensure that the User Name that you are using to login to Discoverer Admin/User editions has necessary privileges assigned by the EUL4_US.

3. Unable to see files under $OKC_TOP/patch/115/discover/US/ after the install of the Discoverer patch ARU for bug 1960714.

   Solution: The OKC related five eex files should be in the new GSCC standard:

   - $<PROD_TOP>/admin/import/US
   - $<PROD_TOP>/patch/..../import/US

   This is because of the recent change in the directory location of the files.

4. Unable to find workbook named: ... Select a valid workbook from the list below...

   Solution: There is a AOL security and authentication related change recently in 11.5.6. With this change, the workbook is now referred by developer key. The d4i workbooks are defined as form functions in AOL, with the workbook name as parameter. There is a change in this. For example:

   - Before: viewer=Y&workbook=Contracts+Listing
   - After: viewer=Y&workbook=OKC_CONTRACT_LISTING

   In case of encountering this error, modify the function parameter values.

3.18 Defining a Process

Use this section to define a system routine procedure that is to be invoked by one of the following:

- Outcome
- Function
- Quality Assurance
- Auto Numbering
- Change Request Process
- Approve, Renew, or Terminate at the appropriate time

A process may be either a Oracle Workflow processes or procedure, either packaged or standalone defined in the parameters.

Use this form to define a source for Process Definition Parameters.

If you define your own procedures or functions, these will not be supported. When you need support, the support representative will request you reproduce the errors without using your procedures.

Prerequisites

- You must create the Oracle Workflow processes or procedures and packages first.
- To use a source for Process Definition, it must first be created. See, *Integrating Source as a JTF Object*.

Steps

1. From the Contracts Manager responsibility, choose Setup > Contract > Process Definition. The Process Definitions window opens.
2. Enter a unique name.
3. Enter the description.
4. Select a purpose from the LOV.
5. Select a type of process from the LOV.
   - If the type is PLSQL, then enter the names for Package and Procedure.
   - If the type is Workflow, then enter the Workflow Name and Workflow Process.
6. Click Validate.
7. From the Basic tab, enter or select the following attributes:
   - Name
   - Data Type
   - Description
8. From the Advanced tab, enter the following attributes:
   - Object Name: (Optional) Enter an appropriate object source. This defines the source for lists of values for Function and Outcome parameters in the Condition form.

- Column Name: (Required when Object Name is used) This allows users to select a column from the specified source as the parameter values for function and outcome for a condition line.
- Description Column: (Optional) Allows users to add description column to the dynamically created LOV used when creating condition lines.

9. Close the form and save your work.

Guidelines

The description appears in the list of values in windows where you need to select a defined process.

You can select one of the following purposes:

- API (Application Programming Interface)
- Contract Approval process
- Change Request process
- Function, for condition evaluations
- Outcome, for Events
- Quality Assurance

When entering a workflow or package and procedure names, make sure the spelling is accurate. The Process Definitions window does not validate the existence or the valid status of the referred objects.

When defining outcomes for events, you cannot use the type script. Script is not referring to a SQL script, but is referring to an internal scripting tool used for Oracle Relationship Planner.

When adding parameters, make sure that they are entered in the correct order.

3.19 Defining Quality Assurance (QA) Checklist

Use this procedure to define a Quality Assurance (QA) checklist. You can designate only one QA checklist as the default checklist for each application. A contract is validated using a QA checklist before submitting for approval. Each checklist consists of one or more Oracle Workflow processes.

Prerequisites

You must define the Oracle Workflow processes before adding them to the checklist.

Steps

1. From the Contracts Manager responsibility, choose Setup > Contract > Quality Assurance.
2. Enter a name and a description.
3. The Default Check List checkbox, if selected, identifies the current QA checklist as the default for the application. You cannot create or delete a default checklist.

   Note: The OKC QA check API includes the OVERRIDE_FLAG parameter. If the flag is set to Yes, the application default and class operation process are not executed. If the flag is set to No (default value), all checks, if applicable, are executed.
4. In the Processes region, select the process that will become a part of the QA checklist.
5. Make sure the Active check box is selected.
6. From the Severity list, select one of the following levels:
   - **Warning:** The contract passes the Approval.
   - **Stop:** The contract does not pass Quality Assurance if this process fails
7. From the Access Level LOV, select one of the following levels:
   - User: Users can always access all User created processes.
   - System: Users cannot update or delete System level processes.
   - Extensible: Users can only activate or deactivate Extensible processes or change their severity level.
8. Optionally, override the default values for the parameters.
9. Save your work.

Guidelines

The default Quality Assurance (QA) checklist is executed automatically for any contract, even if you create another checklist for the contract. You cannot modify or update the default checklist.

3.20 Setting up Actions

Use this form to define or modify actions. Examples of actions include Contract Signed and Counter Group XX Updated.

Define Actions

When an action attribute is defined, the source may also be specified. The action attribute is used for the list of values found within the Expression tab of the Condition form. The source is then used in the Condition form to create list of values for the Right Value field of the Expression tab.

Display Contract Number as Document Source of Task

Actions are related to a document (e.g. a contract). In order to specify the document source, the user must enter a value in the Actions form (at the definition level) to specify the source. Sources are created from JTF Objects. If an appropriate object has not been created, the user must create a new object.

Specify Document Source

Within the Actions form, users can choose the attributes to be used to display the source document number by selecting the Document Number check box. For Contracts this would be a combination of the contract number and contract number modifier. When a task is created from a condition, this enables the display of the contract numbers and contract modifiers in the Source Document Number field of the CRM Administrator Task window.

Note: By default, seeded actions also have the Document Source and Source Document number seeded as well.

Prerequisites

To use a source for the Action Attributes it must first be created. See *Integrating the View as a JTF Object*

Steps

1. From the Contracts Manager responsibility, choose Contract Events > Define Action. The Action form opens.
2. Enter a unique name.
3. Select Action type from the LOV.
4. Enter a description.
5. Enter the Correlation. Correlation is a term used in Advanced Queueing to uniquely identify the type of message in the queue.

6. If you want to enable the Counter tab for this action when defining conditions, then select Counter Action.
7. If you want allow the On-line Action Handling function, then select the Allow Synchronous Outcomes checknox. If you want Asynchronous, or Batch Mode Action Handling, then clear the check box.
8. From the Basic tab enter or select the following attributes. These attributes can be accessed in the condition definition:
   - Name
   - Element Name
   - Description
   - Select Data Type from LOV.
   - If you want a document source, select the Document Number check box for the following attributes:
     - Contract Number Modifier
     - Contract Number
9. From the Advanced tab you can specify the source object for the action attribute:
   - Min/Max Value: (Optional) Validates the range of entries in conditions.
   - Object Name: (Optional) This field is the key to identifying the source for an action attribute. This defines the source for the list of values for the Expression tab in the Condition form.
   - Column Name: Allows the user to select a column from the specified source as list of values for the Right Value of Expression for a condition line.
   - Description Column: Allows the user to add optional description column to the dynamically created LOV used when creating condition lines.
   - Date of Interest: Can be used for action based actions only. Select an attribute when you want the condition to evaluate to true, for example, the entered signing date. If you leave the Date of Interest clear, then the signing date refers to the date when the contract signing date was entered, ignoring the value of the date entered.
   - Used in Conditions: Clear this check box if the attribute should not be used in conditions.
10. Close the form and save your work.

Guidelines

The description of the action appears in the Conditions window.

Select the Enabled check box when you are ready to use the condition, or clear the Enabled check box to disable the condition.

See Also

Integrating the View as a JTF Object

Defining a Process

Defining Condition Templates

3.21 Defining Condition Templates

Use condition templates to define multiple samples of conditions. The templates can then be used to define independent conditions or conditions attached to a contract. Use this procedure to define a condition template.

Prerequisites

Before an outcome can be assigned to a condition template, it must be defined using the Process Definitions form.

Steps

1. From the Contracts Manager responsibility, choose Contract Events > Define Condition Template. The Condition Template window opens.
2. Enter a name and a description.
3. Check the Create a Task check box, if you want to create a task for the Schedule tab in the execution overview.
4. Enter the task owner.
5. Select Action or Date.
6. If you are creating an action condition, then select an action.
7. If you are creating a date condition, then enter the date information.
8. Build your condition lines.

Note: When using the "LIKE" operator, you must not use quotes. For example, when building a condition, where contract_number like "%-2000%", your right value must be %-2000 and not "%-2000".

9. If you want to enter fixed values for parameters for a function, then click Parameters and enter the information.
10. Click Show Condition to display all condition lines and check their syntactical validity.

    If the condition has validated successfully, then Condition Valid is automatically selected.
11. Select **Outcomes.**
12. If you want to assign fixed values or action attribute values to the outcome parameters, then click **Parameters** and enter the information.
13. Select **Notifications.**
14. Select a Success Notifier from the LOV.
15. Select a Failure Notifier from the LOV.
16. Save your work.

Guidelines

You can save your template at any time. You are not required to complete all information in the window.

An independent condition does not need to be tied to a contract and can run independent of a contract. Therefore, an independent condition has to be validated before you can save it.

For the condition of type action, choose if the event is evaluated only once or each time the condition is met. For example: Select the Evaluate Once Only option if you want to evaluate the condition to true only if the counter exceeds 1,000 for the first time. Clear the check box if you want the condition to evaluate to true each time the counter reaches 1,000.

Each condition can consist of one or more condition lines. The sequence determines the order in which the lines are processed. You can enter three types of lines:

- **Expressions**: The left value only allows you to enter attributes defined in the action attributes.
- **Counter**: Select Product or Service and choose a description of the service or product (The Counter tab does not show for Contracts Core).

> **Note:** If you select a counter group, then your left value lists counter group types. If you choose a counter group master, then your left value shows the related counter groups.

- **Functions**: Select Functions declared in the Process Definitions Window. To enter values for parameters, select the **Parameter** button.

3.22 Setting up Approvers

Use this procedure to define the approver for both change requests and contracts if you do not have an Oracle Workflow process defined to handle approvals.

Prerequisites

You do not have an approval process defined using Oracle Workflow.

Steps

1. To set up an approver for change requests, enter the names in the OKC: Change Request Approver system profile option. This profile options overrides a workflow setup and uses the approver, if the workflow is not yet defined.
2. To set up contract approvers, enter the names in the OKC: Contract Approver system profile option. This profile options overrides a workflow setup and uses the approver, if the workflow is not yet defined.

4

Customizing Contracts Core

This chapter covers the following topics as they relate to customizing Contracts Core:

- Customizing Events
- Customizing Workflows
- Registering a New Source as a JTF Object

4.1 Customizing Events

If you customize events, you are doing so at your own risk. Oracle Support can not help you create or debug actions, functions, or outcomes you have created. If you encounter problems with customizations and wish to obtain support, then you must recreate the problem using standard Oracle objects.

4.1.1 Coding Action Assemblers

In the Contracts business process, whenever a defined action occurs, the necessary information about the attributes needs to be assembled and placed into the Events queue. You need to code an action assembler procedure for new actions.

Following are two examples of how you code an action assembler:

Example 1 - When All the Attribute Information is Readily Available

This example accepts all attribute values as parameters. It builds the message table with the attribute name (element_name) and value (element_value). It also creates the record containing correlation for the action.

Then it calls the SEND_MESSAGE Contracts Advanced Queuing procedure to queue the message.

This example fits when the action occurs in a form where the form contains all the required information.

Pseudo Code

```
/*=======================================================+
              Copyright (c) 1999 Oracle Corporation
                Redwood Shores, California,USA
                 All rights reserved.

=======================================================*/

SET VERIFY OFF

WHENEVER SQLERROR EXIT FAILURE ROLLBACK;

CREATE OR REPLACE PACKAGE OKC_MY_ASMBLR_PVT AS

g_pkg_name CONSTANT varchar2(100) := 'OKC_MY_ASMBLR_PVT';


PROCEDURE acn_assemble(
  /* mandatory parameters do not change */
  p_api_version       IN NUMBER,
  p_init_msg_list     IN VARCHAR2 DEFAULT OKC_API.G_FALSE,
  x_return_status     OUT NOCOPY VARCHAR2,
  x_msg_count         OUT NOCOPY NUMBER,
  x_msg_data          OUT NOCOPY VARCHAR2,

 /* parameters specific to this action assembler to be added here */
  p_attribute1_value    IN DATATYPE,
  p_attribute2_value    IN DATATYPE,
  . . .
  . . .                              );

END OKC_MY_ASMBLR_PVT;
/

COMMIT;
EXIT;



/*=======================================================+
```

```
                Copyright (c) 1999 Oracle Corporation
                  Redwood Shores, California,USA
                   All rights reserved.

=======================================================*/

SET VERIFY OFF

WHENEVER SQLERROR EXIT FAILURE ROLLBACK;

CREATE OR REPLACE PACKAGE BODY OKC_MY_ASMBLR_PVT AS

PROCEDURE acn_assemble(
  p_api_version       IN NUMBER,
  p_init_msg_list     IN VARCHAR2 DEFAULT OKC_API.G_FALSE,
  x_return_status     OUT NOCOPY VARCHAR2,
  x_msg_count         OUT NOCOPY NUMBER,
  x_msg_data          OUT NOCOPY VARCHAR2,

/* parameters specific to this action assembler to be added here */
  p_attribute1_value    IN/OUT DATATYPE,
  p_attribute2_value    IN/OUT DATATYPE,
  p_attribute3_value    IN/OUT DATATYPE,
  . . .
  . . .                                )  IS

 l_api_name     CONSTANT VARCHAR2(30) := 'acn_assemble';
 l_api_version   CONSTANT NUMBER := 1.0;
 i               NUMBER := 1;
 l_corrid_rec    okc_aq_pvt.corrid_rec_typ;
 l_msg_tbl       okc_aq_pvt.msg_tab_typ;
 l_msg_count     number;
 l_msg_data      varchar2(1000);
 l_return_status varchar2(1);

 /* 'MY_ACTION_CORRELATION' is the correlation of the new action for
 which this assembler is being coded */

 CURSOR cur_corr_csr IS
 SELECT aae.element_name,
        aae.format_mask format_mask
   FROM okc_actions_b acn,
        okc_action_attributes_b aae
  WHERE acn.id = aae.acn_id
    AND acn.correlation = 'MY_ACTION_CORRELATION' ;
```

```
BEGIN

 l_return_status := OKC_API.START_ACTIVITY
                    (l_api_name
                    ,p_init_msg_list
                    ,'_PROCESS'
                    ,x_return_status);

  IF l_return_status = OKC_API.G_RET_STS_UNEXP_ERROR THEN
     RAISE OKC_API.G_EXCEPTION_UNEXPECTED_ERROR;
  ELSIF l_return_status = OKC_API.G_RET_STS_ERROR THEN
     RAISE OKC_API.G_EXCEPTION_ERROR;
  END IF;

l_msg_tbl := okc_aq_pvt.msg_tab_typ();
 FOR corr_rec IN cur_corr_csr
 LOOP
 /*   'ATTRIBUTE1_ELEMENT_NAME' is the 'element name' of the action attribute
for the newly defined action */
 IF corr_rec.element_name = 'ATTRIBUTE1_ELEMENT_NAME' THEN
  l_msg_tbl.extend;
  l_msg_tbl(i).element_name     := corr_rec.element_name;
  l_msg_tbl(i).element_value    := p_attribute1_value;
 ELSIF corr_rec.element_name = 'ATTRIBUTE2_ELEMENT_NAME' THEN
   l_msg_tbl.extend;
  l_msg_tbl(i).element_name    := corr_rec.element_name;
  l_msg_tbl(i).element_value   := p_attribute2_value;

/* If the attribute is of date or number datatype and the datatype is specified
while defining the attribute then do the following */

 ELSIF corr_rec.element_name = 'ATTRIBUTE3_ELEMENT_NAME' THEN
   l_msg_tbl.extend;
   l_msg_tbl(i).element_name :=  corr_rec.element_name;
   IF corr_rec.format_mask IS NOT NULL THEN
     l_msg_tbl(i).element_value := to_char(p_attribute3_value,
                                     corr_rec.format_mask);
   ELSE
     l_msg_tbl(i).element_value := p_attribute3_value;
   END IF;

 ELSIF . . .
  . . .
  . . .
```

```
 END IF;

  i := i + 1;
 END LOOP;

 l_corrid_rec.corrid := 'MY_ACTION_CORRELATION' ;

/* This procedure puts the message on the events queue */
  OKC_AQ_PUB.send_message(p_api_version     =>'1.0'
                         ,x_msg_count       => l_msg_count
                         ,x_msg_data        => l_msg_data
                         ,x_return_status   => l_return_status
                         ,p_corrid_rec      => l_corrid_rec
                         ,p_msg_tab         => l_msg_tbl
                         ,p_queue_name      =>
                         okc_aq_pvt.g_event_queue_name);

    IF l_return_status = OKC_API.G_RET_STS_UNEXP_ERROR THEN
        RAISE OKC_API.G_EXCEPTION_UNEXPECTED_ERROR;
    ELSIF l_return_status = OKC_API.G_RET_STS_ERROR THEN
        RAISE OKC_API.G_EXCEPTION_ERROR;
    END IF;

 OKC_API.END_ACTIVITY(x_msg_count, x_msg_data);

EXCEPTION
 WHEN OKC_API.G_EXCEPTION_ERROR THEN
  x_return_status := OKC_API.HANDLE_EXCEPTIONS
                       (l_api_name,
                        G_PKG_NAME,
                       'OKC_API.G_RET_STS_ERROR',
                        x_msg_count,
                        x_msg_data,
                        '_PROCESS');
 WHEN OKC_API.G_EXCEPTION_UNEXPECTED_ERROR THEN
  x_return_status := OKC_API.HANDLE_EXCEPTIONS
                       (l_api_name,
                        G_PKG_NAME,
                       'OKC_API.G_RET_STS_UNEXP_ERROR',
                        x_msg_count,
                        x_msg_data,
                        '_PROCESS');
 WHEN OTHERS THEN
  x_return_status := OKC_API.HANDLE_EXCEPTIONS
                       (l_api_name,
```

```
                        G_PKG_NAME,
                        'OTHERS',
                        x_msg_count,
                        x_msg_data,
                        '_PROCESS');
END acn_assemble;


END OKC_MY_ACTION_ASMBLR_PVT;
/

COMMIT;
EXIT;
```

Example 2 - When Attribute Information Needs to Be Assembled

This example accepts key information about the action, for example contract_id, and collects the values for all the attributes from the database. It builds the message table with the attribute element name (element_name) and value (element_value). It also creates the record containing correlation for the action.

Then it calls the SEND_MESSAGE Contracts Advanced Queuing procedure to queue the message.

This example fits where the action occurs in the middle of a workflow process or when information is not readily available.

Pseudo Code

```
/*=======================================================+
|               Copyright (c) 1999 Oracle Corporation
|                  Redwood Shores, California, USA
|                       All rights reserved
+=======================================================*/
SET VERIFY OFF
WHENEVER SQLERROR EXIT FAILURE ROLLBACK;

CREATE OR REPLACE PACKAGE OKC_MY_ACTION_ASMBLR_PVT AS
g_pkg_name     CONSTANT varchar2(100) := 'OKC_MY_ACTION_ASMBLR_PVT';

---------------------------------------------------------------------
-- PROCEDURE acn_assemble
---------------------------------------------------------------------
  PROCEDURE acn_assemble(
/* mandatory parameters do not change */
```

```
    p_api_version                 IN NUMBER,
    p_init_msg_list              IN VARCHAR2 DEFAULT OKC_API.G_FALSE,
    x_return_status               OUT NOCOPY VARCHAR2,
    x_msg_count                   OUT NOCOPY NUMBER,
    x_msg_data                      OUT NOCOPY VARCHAR2,
/* parameters essential to gather attribute values for this action to be added
here */
    p_my_param                   IN/OUT DATATYPE,
    . . .);
END OKC_MY_ACTION_ASMBLR_PVT;
/
commit;
exit;

/*=======================================================+
|               Copyright (c) 1999 Oracle Corporation
|                  Redwood Shores, California, USA
|                       All rights reserved.
+=======================================================*/

SET VERIFY OFF
WHENEVER SQLERROR EXIT FAILURE ROLLBACK;

CREATE OR REPLACE PACKAGE BODY OKC_MY_ACTION_ASMBLR_PVT AS
  g_pkg_name     CONSTANT varchar2(100) := 'OKC_MY_ACTION_ASMBLR_PVT';

  PROCEDURE acn_assemble(
/* mandatory parameters do not change */
    p_api_version                  IN NUMBER,
    p_init_msg_list                IN VARCHAR2 DEFAULT OKC_API.G_FALSE,
    x_return_status                OUT NOCOPY VARCHAR2,
    x_msg_count                    OUT NOCOPY NUMBER,
    x_msg_data                     OUT NOCOPY VARCHAR2,
 /* parameters essential to gather attribute values for this action to be added
here */
    p_my_param                   IN/OUT DATATYPE
    . . .) IS
--
    l_api_name            CONSTANT VARCHAR2(30) := 'ACN_ASSEMBLE';
    l_api_version         NUMBER := 1.0;
    l_init_msg_list       VARCHAR2(1) DEFAULT OKC_API.G_FALSE;
    l_return_status varchar2(1) := OKC_API.G_RET_STS_SUCCESS;
--
  /*Declare cursor or cursors to gather attribute values using
   p_my_param */
```

```
    CURSOR my_cur IS
    SELECT attribute1_value,
           attribute2_value,
           . . .,
           . . .
    FROM  table1,
          table2,
          . . .
    WHERE table1.id = p_my_param
    AND   . . .;
    my_rec my_cur%ROWTYPE;

    /* Declare cursor to get all attributes element_names, format masks(for
date,number datatypes) for the newly created action*/
/* 'MY_ACTION_CORRELATION' is the correlation of the new action for
 which this assembler is being coded */

    CURSOR acn_cur IS
    SELECT aae.element_name element_name,
           aae.format_mask format_mask
    FROM   okc_actions_b acn,
           okc_action_attributes_b aae
    WHERE  acn.id = aae.acn_id
    AND    acn.correlation = 'MY_ACTION_CORRELATION';
    acn_rec acn_cur%ROWTYPE;

--
   l_rec okc_aq_pvt.corrid_rec_typ;
   l_tbl okc_aq_pvt.msg_tab_typ;
   i     NUMBER := 1;
--
    BEGIN
    -- call start_activity to create savepoint, check comptability
    -- and initialize message list
       l_return_status := OKC_API.START_ACTIVITY(l_api_name
                                                 ,l_init_msg_list
                                                 ,'_PVT'
                                                 ,x_return_status
                                                 );
    -- check if activity started successfully
       IF l_return_status = OKC_API.G_RET_STS_UNEXP_ERROR THEN
          RAISE OKC_API.G_EXCEPTION_UNEXPECTED_ERROR;
       ELSIF l_return_status = OKC_API.G_RET_STS_ERROR THEN
             RAISE OKC_API.G_EXCEPTION_ERROR;
       END IF;
```

```
    l_rec.corrid := 'MY_ACTION_CORRELATION';
    l_tbl := okc_aq_pvt.msg_tab_typ();

  /*   'ATTRIBUTE1_ELEMENT_NAME' is the 'element name' of the action
  attribute1 for the newly defined action */

    FOR acn_rec IN acn_cur LOOP
      OPEN my_cur;
      FETCH my_cur INTO my_rec;
       IF acn_rec.element_name = 'ATTRIBUTE1_ELEMENT_NAME' THEN
          l_tbl.extend;
          l_tbl(i).element_name  :=  acn_rec.element_name;
        l_tbl(i).element_value := my_rec.attribute1_value;
     ELSIF acn_rec.element_name = 'ATTRIBUTE2_ELEMENT_NAME' THEN
          l_tbl.extend;
        l_tbl(i).element_name  :=  acn_rec.element_name;
          l_tbl(i).element_value := my_rec.attribute2_value;
 /* If the attribute is of date or number datatype and the datatype is specified
while defining the attribute then do the following */
        ELSIF acn_rec.element_name = 'ATTRIBUTE3_ELEMENT_NAME' THEN
          l_tbl.extend;
          l_tbl(i).element_name :=  acn_rec.element_name;
          IF acn_rec.format_mask IS NOT NULL THEN
            l_tbl(i).element_value := to_char(my_rec.attribute3_value,
                                              acn_rec.format_mask);
          ELSE
            l_tbl(i).element_value := my_rec.attribute3_value;
          END IF;

        ELSIF . . .
          . . .
        END IF;
        i := i+1;
      CLOSE my_cur;
    END LOOP;

/* This procedure puts the message on the events queue */
OKC_aq_pvt.send_message(p_api_version   =>  l_api_version
                        ,x_msg_count     => x_msg_count
                        ,x_msg_data      => x_msg_data
                        ,x_return_status => x_return_status
                        ,p_corrid_rec    => l_rec
                        ,p_msg_tab       => l_tbl
                        ,p_queue_name    =>
```

```
                            okc_aq_pvt.g_event_queue_name
                           );
    -- check if activity started successfully
       IF l_return_status = OKC_API.G_RET_STS_UNEXP_ERROR THEN
          RAISE OKC_API.G_EXCEPTION_UNEXPECTED_ERROR;
       ELSIF l_return_status = OKC_API.G_RET_STS_ERROR THEN
          RAISE OKC_API.G_EXCEPTION_ERROR;
       END IF;

    OKC_API.END_ACTIVITY(x_msg_count, x_msg_data);

    EXCEPTION
       WHEN OKC_API.G_EXCEPTION_ERROR THEN
       x_return_status := OKC_API.HANDLE_EXCEPTIONS
       (l_api_name,
        G_PKG_NAME,
        'OKC_API.G_RET_STS_ERROR',
        x_msg_count,
        x_msg_data,
        '_PVT');
       WHEN OKC_API.G_EXCEPTION_UNEXPECTED_ERROR THEN
       x_return_status := OKC_API.HANDLE_EXCEPTIONS
       (l_api_name,
        G_PKG_NAME,
        'OKC_API.G_RET_STS_UNEXP_ERROR',
        x_msg_count,
        x_msg_data,
        '_PVT');
       WHEN OTHERS THEN
       x_return_status := OKC_API.HANDLE_EXCEPTIONS
       (l_api_name,
        G_PKG_NAME,
        'OTHERS',
        x_msg_count,
        x_msg_data,
        '_PVT');
    END acn_assemble;

END OKC_K_MY_ACTION_PVT;
/
commit;
exit;
```

4.1.1.1 Coding Functions for Function Expressions in Condition Lines

The functions used in the Function tab of the Conditions window should always return T or F as a return value. The following example shows how a function expression can be used to determine if a contract belongs to a particular customer. While entering the parameters for the function expression in the Conditions window, the user can enter a value instead of selecting attributes for p_role and p_customer_name parameters . This way users can check if the contract belongs to a customer they are interested in. The condition evaluator executes this function with the user entered values at run time and returns T or F, based on which certain outcomes can be executed.

```
FUNCTION customer_exists(p_kid IN NUMBER,
                         p_customer_name IN VARCHAR2,
                         p_role    IN  VARCHAR2)
RETURN VARCHAR2 IS

CURSOR cust_cur IS
select 'X'
from  okc_k_party_roles_v role,
      okx_parties_v party
where role.object1_id1 = party.id1
and   role.object1_id2 = party.id2
and   role.jtot_object1_code in ('OKX_PARTY','OKX_OPERUNIT')
and   upper(role.rle_code) = upper(p_role)
and   role.chr_id = p_kid
and   party.name = p_customer_name;
cust_rec  cust_cur%ROWTYPE;

BEGIN
   OPEN cust_cur;
   FETCH cust_cur INTO cust_rec;
     IF cust_cur%FOUND THEN
       RETURN('T');
     ELSE
       RETURN('F');
     END IF;
   CLOSE cust_cur;
EXCEPTION
  WHEN others THEN
  RETURN('F');
END customer_exists;
```

4.1.1.2 Coding Outcomes for Conditions

All PL/SQL Outcomes that do not have the standard set of parameters listed below should have a wrapper API that provides these parameters and default values. All outcomes should have p_init_msg_list set to OKC_API.G_TRUE to initialize message stack.

p_api_version	IN NUMBER,
p_init_msg_list	IN VARCHAR2 DEFAULT OKC_API.G_TRUE,
x_return_status	OUT NOCOPY VARCHAR2,
x_msg_count	OUT NOCOPY NUMBER,
x_msg_data	OUT NOCOPY VARCHAR2,

> **Note**, that it is not necessary to define these standard parameters in the Process Definitions form, as the Outcome Initiator will always include these parameters in the procedure call. The Outcome must not contain any nonstandard OUT parameters. If these are included in an outcome it will fail on execution with a 'PLS-00363: expression '<expression>' cannot be used as an assignment target' error.

If there is a nonstandard OUT parameter in the required outcome - a wrapper API will need to be written which includes a local variable where the OUT parameter can be assigned to.

```
EXCEPTION
       WHEN OKC_API.G_EXCEPTION_ERROR THEN
       x_return_status := OKC_API.HANDLE_EXCEPTIONS
       (l_api_name,
        G_PKG_NAME,
        'OKC_API.G_RET_STS_ERROR',
        x_msg_count,
        x_msg_data,
        '_PVT');
       WHEN OKC_API.G_EXCEPTION_UNEXPECTED_ERROR THEN
       x_return_status := OKC_API.HANDLE_EXCEPTIONS
       (l_api_name,
        G_PKG_NAME,
        'OKC_API.G_RET_STS_UNEXP_ERROR',
        x_msg_count,
        x_msg_data,
        '_PVT');
```

```
    WHEN OTHERS THEN
    x_return_status := OKC_API.HANDLE_EXCEPTIONS
    (l_api_name,
     G_PKG_NAME,
     'OTHERS',
     x_msg_count,
     x_msg_data,
     '_PVT');
END;
```

Any messages specific to the outcome should be created in the FND_NEW_ MESSAGES table. If the Outcome is a workflow, then it need not have any out parameters. The errors in Workflow will be handled by the standard Workflow functionality.

4.1.2 Coding Date Based Action Assemblers

Users can define actions that are not triggered by originating actions, but are related to a certain point in time, such as the date when contracts are about to expire, or a customer's anniversary date within the specified period, and so on. To process all the date based actions, a concurrent manager is scheduled to run once everyday to run the Date Assembler concurrent program which checks for all records that satisfy the date condition and builds a message table with all the attributes and values that is put in the queue for evaluation.

Following is an example to code a date based action assembler for contracts that are about to expire. Please note that:

- All these actions need to be defined along with correlation ex: KEXPIRE and attributes that can be derived from the information provided by the action, such as condition header id, contract category, contract class, contract expiry date, contract id, contract number, contract number modifier, contract status, estimated amount, and so on using the Define Action form.
- Using the Define Condition form you define whether the event is a date based event. The exact point in time must be indicated: For example "45 days before contract expiration date".

Refer to Oracle Applications Developer's Guide for information on Concurrent programs. The Concurrent program date assembler scheduled to run once everyday selects a contract or all contracts (for independent conditions) expiring before or after the given number of days from the current date.

Pseudo Code

Date action assembler pub

```
/*=========================================================== Copyright (c)
1999 Oracle Corporation
Redwood Shores, California, USA
All rights reserved.
+===========================================================*/
SET VERIFY OFF
WHENEVER SQLERROR EXIT FAILURE ROLLBACK;
CREATE OR REPLACE PACKAGE BODY MY_DATE_ASMBLR_PUB AS
PROCEDURE conc_mgr(errbuf  OUT VARCHAR2,
                                        retcode OUT VARCHAR2) IS

  l_api_name        CONSTANT VARCHAR2(30) := 'conc_mgr';
  l_api_version     CONSTANT VARCHAR2(30) := 1.0;
  l_return_status   VARCHAR2(1)  := OKC_API.G_RET_STS_SUCCESS;
  x_return_status   VARCHAR2(1)  := OKC_API.G_RET_STS_SUCCESS;
  l_msg_count       NUMBER;
  l_msg_data        VARCHAR2(1000);
  l_init_msg_list   VARCHAR2(3) := 'F';

  BEGIN
       -- call start_activity to create savepoint, check comptability
       -- and initialize message list
       l_return_status := OKC_API.START_ACTIVITY(l_api_name
                                                 ,l_init_msg_list
                                                 ,'_PUB'
                                                 ,x_return_status
                                                 );
    --Initialize the return code
    retcode := 0;
MY_DATE_ASMBLR_PUB.date_expire_asmblr(
           p_api_version    => l_api_version
                 ,p_init_msg_list  => l_init_msg_list
                 ,x_return_status  => l_return_status
                 ,x_msg_count     => l_msg_count
     ,x_msg_data     => l_msg_data);

IF l_return_status = OKC_API.G_RET_STS_UNEXP_ERROR THEN
RAISE OKC_API.G_EXCEPTION_UNEXPECTED_ERROR;
ELSIF l_return_status = OKC_API.G_RET_STS_ERROR THEN
RAISE OKC_API.G_EXCEPTION_ERROR;
ELSIF l_return_status = OKC_API.G_RET_STS_SUCCESS THEN
                commit;
```

```
      END IF;
OKC_API.END_ACTIVITY(l_msg_count, l_msg_data);
EXCEPTION
     WHEN OKC_API.G_EXCEPTION_ERROR THEN
             retcode := 2;
              errbuf  := substr(sqlerrm,1,200);
       l_return_status := OKC_API.HANDLE_EXCEPTIONS
       (l_api_name,
        G_PKG_NAME,
        'OKC_API.G_RET_STS_ERROR',
        l_msg_count,
        l_msg_data,
        '_PUB');
       WHEN OKC_API.G_EXCEPTION_UNEXPECTED_ERROR THEN
retcode := 2;
errbuf  := substr(sqlerrm,1,200);
       l_return_status := OKC_API.HANDLE_EXCEPTIONS
       (l_api_name,
        G_PKG_NAME,
        'OKC_API.G_RET_STS_UNEXP_ERROR',
        l_msg_count,
        l_msg_data,
        '_PUB');
       WHEN OTHERS THEN
retcode := 2;
errbuf  := substr(sqlerrm,1,200);
       l_return_status := OKC_API.HANDLE_EXCEPTIONS
       (l_api_name,
        G_PKG_NAME,
        'OTHERS',
        l_msg_count,
        l_msg_data,
        '_PUB');
  END conc_mgr;

  PROCEDURE date_expire_asmblr(
    p_api_version                  IN NUMBER DEFAULT 1.0,
    p_init_msg_list                IN VARCHAR2 DEFAULT OKC_API.G_FALSE,
    x_return_status                OUT NOCOPY VARCHAR2,
    x_msg_count                    OUT NOCOPY NUMBER,
    x_msg_data                     OUT NOCOPY VARCHAR2) IS

    l_api_name            CONSTANT VARCHAR2(30) := 'date_assemble';
  l_return_status     VARCHAR2(1) := OKC_API.G_RET_STS_SUCCESS;
```

```
    --Get all the action and condition details for a date based event
    CURSOR acn_csr IS
    SELECT acn.correlation,
           acn.acn_type,
           cnh.id,
           cnh.dnz_chr_id,
           cnh.cnh_variance,
           cnh.before_after
    FROM okc_condition_headers_v cnh,
         okc_actions_v acn
    WHERE cnh.acn_id = acn.id
    AND acn.acn_type = 'DBA'
    AND cnh.condition_valid_yn = 'Y'
    AND cnh.template_yn = 'N';

    acn_recacn_csr%ROWTYPE;

    BEGIN
    -- call start_activity to create savepoint, check comptability
    -- and initialize message list
       l_return_status := OKC_API.START_ACTIVITY(l_api_name
                                                 ,p_init_msg_list
                                                 ,'_PUB'
                                                 ,x_return_status
                                                 );
       -- check if activity started successfully
       IF l_return_status = OKC_API.G_RET_STS_UNEXP_ERROR THEN
          RAISE OKC_API.G_EXCEPTION_UNEXPECTED_ERROR;
       ELSIF l_return_status = OKC_API.G_RET_STS_ERROR THEN
             RAISE OKC_API.G_EXCEPTION_ERROR;
       END IF;

--If the action type is date based action and the correlation is Contract
expiry date
--then call the date action assembler to process all the contracts expiring
before or after
--the given number of days
       FOR acn_rec IN acn_csr LOOP
             IF acn_rec.acn_type = 'DBA' and acn_rec.correlation = 'KEXPIRE'
THEN
--Call the date assembler for contract expiry date
    MY_EXP_DATE_ASMBLR_PVT.contract_exp_date_asmblr(
              p_api_version   => 1
                 ,p_init_msg_list        => 'F'
                 ,x_return_status        => x_return_status
```

```
                ,x_msg_count           => x_msg_count
                ,x_msg_data             => x_msg_data
                ,p_cnh_id                 => acn_rec.id
                ,p_dnz_chr_id          => acn_rec.dnz_chr_id
                ,p_cnh_variance       => acn_rec.cnh_variance
                ,p_before_after         => acn_rec.before_after);
       IF x_return_status = OKC_API.G_RET_STS_UNEXP_ERROR THEN
          RAISE OKC_API.G_EXCEPTION_UNEXPECTED_ERROR;
       ELSIF x_return_status = OKC_API.G_RET_STS_ERROR THEN
RAISE OKC_API.G_EXCEPTION_ERROR;
ELSIF x_return_status = OKC_API.G_RET_STS_SUCCESS THEN
commit;
       END IF;
ELSIF  acn_rec.acn_type = 'DBA' and acn_rec.correlation = 'CORRELATION' THEN
--Call the  date assembler for --------------------
  END IF;
      END LOOP;
    OKC_API.END_ACTIVITY(x_msg_count, x_msg_data);

    EXCEPTION
       WHEN OKC_API.G_EXCEPTION_ERROR THEN
       x_return_status := OKC_API.HANDLE_EXCEPTIONS
       (l_api_name,
        G_PKG_NAME,
        'OKC_API.G_RET_STS_ERROR',
        x_msg_count,
        x_msg_data,
        '_PUB');
       WHEN OKC_API.G_EXCEPTION_UNEXPECTED_ERROR THEN
       x_return_status := OKC_API.HANDLE_EXCEPTIONS
       (l_api_name,
        G_PKG_NAME,
        'OKC_API.G_RET_STS_UNEXP_ERROR',
        x_msg_count,
        x_msg_data,
        '_PUB');
       WHEN OTHERS THEN
       x_return_status := OKC_API.HANDLE_EXCEPTIONS
       (l_api_name,
        G_PKG_NAME,
        'OTHERS',
        x_msg_count,
        x_msg_data,
        '_PUB');
    END date_expire_asmblr;
```

```
END MY_DATE_ASMBLR_PUB;
/
commit;
exit;
```

Date Action Assembler

For coding MY_EXP_DATE_ASMBLR_PVT Refer to Coding Action assemblers, example 2 when attribute information needs to be assembled.

```
/*=============================================================+
|        Copyright (c) 1999 Oracle Corporation
|          Redwood Shores, California, USA
|             All rights reserved.
+=============================================================*/

SET VERIFY OFF

WHENEVER SQLERROR EXIT FAILURE ROLLBACK;

CREATE OR REPLACE PACKAGE BODY MY_EXP_DATE_ASMBLR_PVT AS

  PROCEDURE date_assemble(
    p_api_versionIN NUMBER,
    p_init_msg_list     IN VARCHAR2 DEFAULT OKC_API.G_FALSE,
    x_return_status     OUT NOCOPY VARCHAR2,
    x_msg_count         OUT NOCOPY NUMBER,
    x_msg_data          OUT NOCOPY VARCHAR2,
    p_cnh_id               IN NUMBER,
    p_dnz_chr_id        IN IN NUMBER,
    p_cnh_variance     IN IN NUMBER,
    p_before_after       IN VARCHAR2) IS


    l_api_name            CONSTANT VARCHAR2(30) := 'date_assemble';
    l_return_status       VARCHAR2(1) := OKC_API.G_RET_STS_SUCCESS;
    k                     NUMBER := 1;
    l_last_rundate DATE;
    l_variance                 okc_condition_headers_b.cnh_variance%TYPE;
    l_corrid_rec      okc_aq_pub.corrid_rec_typ;
    l_msg_tbl         okc_aq_pub.msg_tab_typ;

    --Get the last rundate of the date assembler concurrent program
    CURSOR last_rundate_csr IS
```

```
    SELECT nvl(max(req.actual_completion_date), (sysdate - 1)) last_rundate
    FROM   fnd_concurrent_programs prg,fnd_concurrent_requests req
    WHERE  prg.application_id = req.program_application_id
    AND    prg.concurrent_program_id = req.concurrent_program_id
    AND    prg.concurrent_program_name = 'CONCURRENT_PROGRAM_NAME'
    AND    req.status_code <> 'E';

    --Get all the contracts that are expiring from the last rundate
    --of the date assembler concurrent program for a given condition
    CURSOR attribute_value_csr(p_chr_id IN NUMBER, p_variance IN NUMBER, p_last_
rundate IN DATE) is
    SELECT a.attr_value1,
           a.attr_value2,
           a.attr_value3,
               -------------
    FROM   table1 a, table2 b
    WHERE (a.id = p_chr_id OR p_chr_id IS NULL)
    AND ------------------------------
    AND trunc(a.end_date) BETWEEN
    (p_variance + trunc(p_last_rundate) + 1)
    AND (trunc(sysdate) + p_variance);

    CURSOR elements_csr IS
    SELECT aae.element_name
    FROM   okc_actions_v acn,okc_action_attributes_v aae
    WHERE  acn.id = aae.acn_id
    AND    acn.correlation = 'CORRELATION_NAME';

    elements_rec elements_csr%ROWTYPE;

    BEGIN
    -- call start_activity to create savepoint, check comptability
    -- and initialize message list
       l_return_status := OKC_API.START_ACTIVITY(l_api_name
                                                 ,p_init_msg_list
                                                 ,'_PVT'
                                                 ,x_return_status
                                                 );
    -- check if activity started successfully
       IF l_return_status = OKC_API.G_RET_STS_UNEXP_ERROR THEN
          RAISE OKC_API.G_EXCEPTION_UNEXPECTED_ERROR;
       ELSIF l_return_status = OKC_API.G_RET_STS_ERROR THEN
             RAISE OKC_API.G_EXCEPTION_ERROR;
       END IF;
```

```
    --Get the correlation
    SELECT acn.correlation INTO l_corrid_rec FROM okc_actions_v acn
    WHERE acn.correlation = 'CORRELATION_NAME';

    --Fetch the last rundate of the date assembler concurrent program
    OPEN last_rundate_csr;
    FETCH last_rundate_csr into l_last_rundate;
    CLOSE last_rundate_csr;

        --Check if the variance is positive or negative
IF p_before_after = 'A' THEN
l_variance := p_cnh_variance * -1;
ELSIF p_before_after = 'B' THEN
l_variance := p_cnh_variance;
END IF;

      --get contract details
      FOR attribute_value_rec in attribute_value_csr(p_chr_id       => p_dnz_
chr_id,
                          p_variance     => l_variance,
                          p_last_rundate => l_last_rundate) LOOP
          k := 1;
   l_contract_number := k1_rec.k_number;
   --Initialize the table
   l_msg_tbl   := okc_aq_pvt.msg_tab_typ();
          FOR elements_rec IN elements_csr LOOP
--Build the elements table
    IF elements_rec.element_name = 'ATTRIBUTE1_ELEMENT_NAME' THEN
    l_msg_tbl.extend;
       l_msg_tbl(k).element_name  := elements_rec.element_name;
      l_msg_tbl(k).element_value := attribute_value_rec.attr_val1;
  ELSIF elements_rec.element_name = 'ATTRIBUTE2_ELEMENT_NAME' THEN
    l_msg_tbl.extend;
       l_msg_tbl(k).element_name  := elements_rec.element_name;
       l_msg_tbl(k).element_value := attribute_value_rec.attr_val2;
  ELSIF elements_rec.element_name = 'ATTRIBUTE3_ELEMENT_NAME' THEN
    l_msg_tbl.extend;
       l_msg_tbl(k).element_name  := elements_rec.element_name;
       l_msg_tbl(k).element_value := attribute_value_rec.attr_val3;
    ELSIF ------------------------------------------------------------
   --Increment the counter
k := k + 1;
END LOOP;

--Call the Advanced queue procedure to put the message on the queue
```

```
OKC_AQ_PUB.send_message(p_api_version      =>'1.0'
              ,p_init_msg_list    => 'F'
                                        ,x_msg_count       => x_msg_count
                                       ,x_msg_data         => x_msg_data
                                       ,x_return_status    => l_return_status
                                       ,p_corrid_rec        => l_corrid_rec
                                      ,p_msg_tab           => l_msg_tbl
                                       ,p_queue_name     => okc_aq_pvt.g_event_
queue_name);
IF l_return_status = OKC_API.G_RET_STS_UNEXP_ERROR THEN
          RAISE OKC_API.G_EXCEPTION_UNEXPECTED_ERROR;
      ELSIF l_return_status = OKC_API.G_RET_STS_ERROR THEN
            RAISE OKC_API.G_EXCEPTION_ERROR;
ELSIF l_return_status = OKC_API.G_RET_STS_SUCCESS THEN
commit;
       END IF;
    END LOOP;

    OKC_API.END_ACTIVITY(x_msg_count, x_msg_data);

    EXCEPTION
       WHEN OKC_API.G_EXCEPTION_ERROR THEN
       x_return_status := OKC_API.HANDLE_EXCEPTIONS
       (l_api_name,
        G_PKG_NAME,
        'OKC_API.G_RET_STS_ERROR',
        x_msg_count,
        x_msg_data,
        '_PVT');
       WHEN OKC_API.G_EXCEPTION_UNEXPECTED_ERROR THEN
       x_return_status := OKC_API.HANDLE_EXCEPTIONS
       (l_api_name,
        G_PKG_NAME,
        'OKC_API.G_RET_STS_UNEXP_ERROR',
        x_msg_count,
        x_msg_data,
        '_PVT');
       WHEN OTHERS THEN
       x_return_status := OKC_API.HANDLE_EXCEPTIONS
       (l_api_name,
        G_PKG_NAME,
        'OTHERS',
        x_msg_count,
        x_msg_data,
        '_PVT');
```

```
    END date_assemble;

END MY_EXP_DATE_ASMBLR_PVT;
/
commit;
exit;
```

4.1.3 Rules For Creating Actions

1. All Actions defined should have at least one attribute.
2. Counter based Actions should have COUNTER_GROUP_ID and COUNTER_GROUP_LOG_ID as mandatory attributes.

4.2 Customizing Workflows

Oracle Workflow enables you to automate business processes, routing information of any type according to business rules. Workflow automates two procedures in Oracle Contracts:

- Customizing Approval workflow
- Contracts Change Requests workflow

You can use Oracle Workflow Builder to customize these workflows.

4.2.1 Customizing Approval Workflow

The Approval Workflow OKCAUKAP is using the process: K_APPROVAL_PROCESS.

The default Approval workflow routes the approval to the user SYSADMIN, see Setting up the Approval Process. After the user SYSADMIN has approved the workflow, the initiator of the workflow receives a notification to sign the contract.

The recommendation is to modify the workflow, so that after the last approval is obtained, the signature is requested. After the contract is signed, it may get the status Active or Signed, depending on the effective dates of the contract.

4.2.2 Customizing Change Requests Workflow

The Change Request Workflow OKCADCRQ is using the process: CHK_APPROVAL_PROCESS. The default approval workflow routes the approval to the user SYSADMIN, see Setting up the Approval Process.

4.3 Registering a New Source as a JTF Object

Oracle Contracts Core allows you to define your own list of values in several forms. These list of values are called sources. You can use these sources when defining:

- Line Style sources
- Role sources.
- Action Attributes
- Process Definitions

The process of making any kind of data available as sources consists of two procedures:

1. Defining a view
2. Integrating the view into as a JTF object

To explain how to register a new source as a JTF object, an example will be shown. Assume that you need a line style source that qualifies frequencies:

Value	Description
Never	Never
Seldom	1 to 4 times
Often	Ranging from 5 to not always
Always	Without a doubt, always happens

4.3.1 Defining a View

To define a view that can be integrated into a JTF object, the view must have the format outlined below:

Name	Nulll	Type
ID1	NOT NULL	NUMBER
ID2	NOT NULL	NUMBER
Name	NOT NULL	VARCHAR2(11)
DESCRIPTION	NOT NULL	VARCHAR2(31)

Name	Nulll	Type
PRIMARY_UOM_CODE	NOT NULL	VARCHAR2(0)
STATUS	NOT NULL	CHAR(1)
START_DATE_ACTIVE		DATE
END_DATE_ACTIVE		DATE

ID1 and ID2 ensure the uniqueness of the rows. If ID1 is selective enough to be unique, you can fill in a dummy character like '#' as ID2.

To define the view OKX_FREQ_V, following the mandatory format, run the following script:

```
CREATE OR REPLACE VIEW okx_freq_v AS
SELECT 1 ID1, '#' ID2, 'Often' NAME, 'ranging from 5 to not always' DESCRIPTION,
'A' STATUS, sysdate START_DATE_ACTIVE, sysdate END_DATE_ACTIVE,
null PRIMARY_UOM_CODE
FROM DUAL
UNION SELECT 2 ID1, '#' ID2, 'Never' NAME, 'Never' DESCRIPTION,
'A' STATUS, sysdate START_DATE_ACTIVE, sysdate END_DATE_ACTIVE,
null PRIMARY_UOM_CODE
FROM DUAL
UNION SELECT 3 ID1, '#' ID2, 'Hardly ever' NAME, '1 to 4 times' DESCRIPTION,
'A' STATUS, sysdate START_DATE_ACTIVE, sysdate END_DATE_ACTIVE,
null PRIMARY_UOM_CODE
FROM DUAL
UNION SELECT 4 ID1, '#' ID2, 'Always' NAME,
'Without a doubt, always happens' DESCRIPTION,
'A' STATUS, sysdate START_DATE_ACTIVE, sysdate END_DATE_ACTIVE,
null PRIMARY_UOM_CODE
FROM DUAL
```

Note: For line items, you need the column primary_uom_code only if you want to price the line.

4.3.2 Integrating the View as a JTF Object

This procedure gives you an example for the typical case of registering a view as a JTF object. You can also use this procedure for creating a source for Action Attributes and Process Definition Parameters.

ReferenceS

Oracle CRM Application Concepts and Procedures, Task Manager

Steps

1. From CRM Administrator responsibility, navigate to Task and Escalation Manager > Setup > Objects Meta-data. The Task Setup: Object Types window OPENS. See below table for example values.
2. Enter a name
3. Enter a description.
4. Enter a unique object code that will be used for internal identification.
5. Enter a start date.
6. In the Select Statement Details section, enter a value for the From field.
7. Enter a value for the Where field.
8. In the Order by field, enter the object code name and column name you want to order by.

Field	Example Values
Name	Frequency
Description	Frequency in non quantitative terms
Object Code	OKX_FREQ
Start Date	Today's date
From (in the Select Statement Details)	OKX_FREQ_V
Where	OKX_FREQ
Order by (in the Select Statement Details)	OKX_FREQ.name

9. Select an Object User fromthe LOV. The following usages are representative for Contracts Core:

Object User	Description
OK	Contracts

Object User	Description
OKC_EVENTS	Contracts events
OKX	Contract source
OKX_B_O_ROLE	Object constrained for buy
OKX_B_S_ROLE	Subject constrained for buy
OKX_CONTACTS	Contract contact source
OKX_ CURRENCY	Constraint by currency
OKX_LINES	Contract line style source
OKX_ROLES	Contract role source
OKX_RULES	Contract rule source
OKX_S_O_ROLE	Object constrained for sell
OKX_S_S_ROLE	Subject constrained for sell

Note: Despite the fact that you can add more Object users, adding Object users beyond the users listed here will not benefit you. Only the seeded values support the standard forms and functionality.

10. If you want to include your new List of Values in the linestyles, you have to register the line style as an object user. Enter the OKX_LINES as the object user.
11. Close the form and save your changes.

Guidelines

When creating a source use the following guidelines:

- Give an appropriate name and description to the object.
- Object Name OKC_xxxxx where xxxxx is an appropriate name for an Action Attribute Source.
- Provide a minimum of "From Table" and if necessary a "Where" clause and "Order By". Columns will be defined in the "Define Actions" and "Process Defintions" forms.

See Also

Setting Up Actions

Defining a Process

5

Integrating Contracts Core with Oracle Pricing

5.1 Setting up Pricing Integration Overview

With integration to Oracle Pricing the Contracts user can access current discounts that an organization might be offering at that time and include those in the contract without having to perform manual calculations. Other adjustments such as surcharges can be applied automatically as well.

With the integration to Oracle Pricing you can do the following:

- Assign pricelist at the contract header
- Assign pricelist at the line level
- Use current discounts or surcharges defined in Oracle Pricing for the items in the contract
- Override the existing price
- Calculate the price for priced lines

This chapter covers the following topics:

- Setting Up Pricing for Determinant Component
- Troubleshooting Pricing Integration

5.1.1 Prerequisite Setups in Oracle Pricing

Refer to the O*racle Pricing User's Guide* for setup details. In summary the steps would include:

1. Defining price lists and modifier lists.
2. Defining new qualifier and attribute texts (other than seeded values).
3. Defining new pricing contexts and attributes (other than seeded values).
4. Defining attribute mapping between linestyle, rules, party roles (if different then seeded values). The mapping for pricing free format lines and customer items might have to be done at the customer site.
5. Changing search flag or user search flag value on seeded events phases for the Batch Pricing event.

Guidelines

Contracts Core does not mandate a pricelist be attached to an item or contract before Pricing is called. If you attach a pricelist, then that pricelist is called. If you do not attach a pricelist, then you could get the "pricelist not found" message. To avoid this, you can have Pricing return the most qualifying pricelist. This is done by setting the User Search field to Yes for "list Price" within the Events Phases window of Oracle Pricing.

5.2 Pricing Upgrade Concurrent Program

Oracle Contracts has a tight integration with Oracle Pricing product. For the purpose of integration there were data model changes in the contract lines and headers table. The purpose of the Oracle Pricing Data Migration Concurrent Program is to migrate contracts related data of customers prior to 11.5.6 to the new data model changes introduced in 11.5.6 and to enable the Advanced Pricing Profile.

The following describes the parameters of this concurrent program:

- Default Price List: If the priced lines do not have any pricing rules attached to them or any level above them and if there is no pricing rule attached at the contract header, the price_list_id for the priced lines will be defaulted to the

user entered p_dflt_price_list_id parameter. The Default Price List is a required parameter.

- Category: You can specify the category of the contracts to upgrade. If you do not specify the category, the concurrent program will fetch all categories of classes belonging to Contracts Core or Contracts for Sales and run the upgrade. At any given point of time, only one concurrent program can be in progress for a category.
- Enable Advanced Pricing Profile: After all the contracts have been upgraded and you have ran the exceptions scripts and verified the report, you can use this option to enable the OKC Advanced Pricing profile option. If you select this option, it will check if all the categories have been successfully upgraded. If yes, it will update the SUMMARY record in okc_qp_upgrade to complete and set the OKC_ADVANCED_PRICING profile to Y. This option can only be used only after all the contracts have been successfully upgraded
- Upgrade Status Report: This report shows all the categories that have been upgraded and identifies any categories that are not yet upgraded.

References

Oracle Pricing User's Guide, Creating a Modifier.

Prerequisites

Before running the Upgrade program a modifier must be created. This enables modifier details to create manual adjustments for all priced lines if there is any difference between line_list_price and price_negotiated.

Use the Define Modifier form by navigating to Setup > Others > System > Pricing > Modifier Setup for creating the modifier. The information you need to create the modifier is as follows:

Modifier name: OKC_QP_UPGRADE.

Modifier Summary Tab:

- The modifier must have two line types: Discount and Surcharge. Discount would be used if line_list_price is more than price_negotiated. Surcharge would be used if line_list_price is less than price_negotiated.
- Both the modifier lines must be of type 'Override' and have Pricing Phase as 'List Line Adjustment'.

Discount / Charges Tab:

- Application Method for both lines must be 'Amount'.

  Navigate to Setup > Contract > Categories and Sources > Define Line Styles to select the Item to Price check box. Identify a user defined line style and select the Item to Price check box.

Steps

Caution: When the Pricing Upgrade is in progress, no one should be using Oracle Contracts.

1. From the Contracts Manager responsibility navigate to Control > Requests > Run.
2. Select Upgrade Contracts to Use Advanced Pricing for name. The Parameters form opens.
3. Select the default price list.
4. Select the contract category. If you leave it blank, all categories are selected for Contracts Core and Contracts for Sales.
5. Select Yes for Enable Advanced Pricing (optional). After the contracts have been upgraded, you can use this option to enable the OKC Advanced Pricing profile.
6. Select Yes for Report Upgrade Status (optional). This report shows all the categories that have been upgraded and identifies any categories that are not yet upgraded.
7. Click OK.
8. Click Submit and note ID Request Number.
9. Navigate to Control > Requests > View. Check the phase and status of your request.
10. If you selected Yes for Report Upgrade Status then click View Output to view the report.

5.3 Profile Options

The following Contracts Core profile options are set for integration with Oracle Pricing:

Option	Description	Level
OKC: Enable Advanced Pricing	This profile is used to indicate whether the current user would like to use Oracle Pricing for price related calculation.	Site or Application
OKC: Manual Price Adjustment	This enables you to manually adjust the final price of a contract line or header. The default is "Yes".	Application, Responsibility, or User

5.4 Set Up Pricing for Inventory Items

Through this setup, the items you price come from Oracle Inventory. You need to let Oracle Contracts know which line is holding the item that is being priced. This is accomplished by selecting the Item to Price check box in the Line Styles form. You should be familiar with the following terms:

- Priced: The priced (P) flag indicates the lowest denominator in the list of attributes for a priced item (PI), based on which the price of the priced item (PI) can vary. In the Authoring form, the lines corresponding to priced (P) line styles will be the one holding the price for the Item to Price (PI).
- Item to Price: The item that you wish to price. This is used to indicate the line style which will reflect the item for which the price should be calculated.

Steps

1. From the Contracts Manager Responsibility navigate to Setup > Contract > Categories and Sources > Define Line Styles.
2. Select the line style that you want to set the Item to Price.
3. Select the Item to Price check box.
4. Save changes.

Guidelines

There are many business rules that are considered:

- The Item To Price check box can be set for a linestyle that is above or at the same level for the Priced check box.
- The Item To Price check box can be set for the line styles that have a source with usage "OKX_MTL_SYSTEM_ITEM".
- In a hierarchy of line styles, only one line style can be marked as Item To Price.

- The "Item To Price" is copied to the line's attributes when a contract is authored and this line is used. So, later, in the line styles, if this flag is moved to some other line style, it will not affect the behavior of already authored lines.
- For a line style to be used for Configured item, all the three flags (P, PI, R) should be selected on the same linestyle

5.5 Set Up Pricing for Free Format Lines

In Oracle Contracts Core, free format line styles are used so that you can enter a price for any priced line. For the item that a price is entered may or may not belong to inventory or any other source. Oracle Contracts provides two options to handle pricing of such lines:

5.5.1 Set up Without Pricelist

Pricing without using Pricelist for such cases, you can use the ALL_ITEMS feature in Oracle Pricing where if there is a free format line style in the hierarchy above a priced linestyle and no item with a price in a pricelist is used, a generic unit price will be returned. You can then override the net price at the priced line. The catch here is that the priority of this ALL_ITEMS should be low (very high precedence number) so that this is the last qualifying price to be picked.

References

Oracle Pricing User's Guide, Pricelists

Steps

1. From the Oracle Pricing Manager responsibility, navigate to Price Lists > Price List Setup > List Lines tab.
2. Enter the Price List Name.
3. Select the Active check box.
4. Select a currency.
5. Select Item for Product Context.
6. Select ALL_ITEMS for Item Context.
7. Select ALL for Product Value.
8. Choose a UOM code.
9. Select the Primary UOM check box.

10. Enter "0" for Value.
11. Enter the highest precedence number. You want this price to be returned only if no price can be determined.
12. Save your work.

5.5.2 Set up With Pricelist

If there are certain items that do not come from a table or a view and you still want to price them not once but many times in the contract and would like to specify a unit price for them, you can use this option. The advantage of this option is, even if you are trying to price items as you go, you can fix a unit price for them one time through pricelist and keep using it without having to reenter the unit price every time.

5.5.2.1 Define a Value Set

Following these steps you will define a varchar2 valueset type, or use an existing valueset type and define the item names that you want to price.

References

Oracle Applications Developer's Guide, Defining Value Sets

Steps

1. From the Application Developer responsibility, navigate to Application > Validation > Values > Values, Effective.
2. Enter a name for Value Set.
3. Enter a value.
4. Enter a description.
5. Select the Enabled check box.
6. Save your work.

5.5.2.2 Define Segment for Context Item

Follow these steps to define the Free Format segment for pricing context ITEM.

References

Oracle Applications Flexfield Guide.

Steps

1. From the System Administration Responsibility, navigate to Application > FlexField > Descriptive > Segments.
2. Run query for Oracle Pricing and Pricing Contexts for Application and Title.
3. Select the ITEM in the Context Field Values region.
4. Click Segments. The Segments Summary window opens.
5. From the Segments Summary window, click New. The Segments window opens.
6. In the Segments window, enter FREE FORMAT.
7. Enter a Description (Optional).
8. Select Pricing AttributeXX for Column.
9. Enter a number. This can be sequential with the other Segments.
10. Click Value Set.
11. Select "char_only".
12. Click Value Sets.
13. Close the Value Sets window. You should see your new segment in the Segments Summary window.

5.5.2.3 Map Item Attribute

References:

Oracle Pricing User's Guide, Default Attribute Sourcing Rules

Steps

1. From the Pricing Manager Responsibility, navigate to Setup > Attribute Mapping.
2. Run a query for Line Pricing Attrs.
3. Select PRICING_ATTRIBUTEXX.
4. Click Defaulting Rules. The Attribute Defaulting Rules window opens.

5.5.2.4 Defining Function Returning Attribute Value

Follow these steps to map Pricing_attributeXX, the pricing segment you just defined, to jtot_object_code OKX_FREE. This tells Oracle Pricing that if, it finds a linestyle with this source in the populated global table, then treat the value of it as the value for Pricing_attributeXX (FREE_FORMAT) for pricing context "ITEM". For this, you are using a predefined OKC function defined in package OKC_PRICE_PUB that takes in the name of the global table which will hold the value at runtime and the OKX source against which the value will be held. The global table G_LSE_TABLE in this case is predefined and is used for seeded mappings. This mapping is exactly similar to other seeded mappings for line styles except the source has been changed to 'OKX_FREE' - the hypothetical source for free format lines.

References

Oracle Pricing User's Guide, Default Attribute Sourcing Rules

Steps

1. From the Defaulting Conditions region, enter "1" as Precedence.
2. Select "OKC Line Item Context" for Defaulting Condition.
3. Select the Enabled check box.
4. In the Default Sourcing Rules region, enter "1" for Sequence.
5. Select PL/SQL API as the Source Type. The Defaulting Rules Flexfield window opens.
6. Enter "OKC_PRICE_PUB" as the Package Name.
7. Enter "Get_LSE_SOURCE_VALUE(OKC_PRICE_PUB.G_LSE_TBL, 'OKX_FREE')" for Function Name.
8. Click OK.

5.5.2.5 Define Free Format Item in Pricelist

Following this procedure is mandatory before rebuilding sourcing rules as the new version of the concurrent program for building sourcing rules takes only those mapping into consideration that are being used somewhere either in modifier or Pricelists. This improves the performance of Pricing.

Steps

1. From the Oracle Pricing Manager Responsibility, navigate to Price Lists > Price List Setup > List Lines.
2. Run a query for a price list or create a new price list.
3. Select "Item" for Product Context.
4. Select "FREE_FORMAT" for Product Attribute.
5. Select a Product Value from the LOV.
6. Select a UOM.
7. Select the Primary UOM check box.
8. Enter a Value.
9. Save your work.

5.5.2.6 Build Sourcing Rules

Follow this procedure for submitting a concurrent request to run "Build Sourcing Rules"

Steps

1. From the Oracle Pricing Manager Responsibility, navigate to View Concurrent Requests.
2. Click Submit a New Request.
3. Select the Build Sourcing Rules option.
4. Click Submit.
5. Run query for your Request ID.
6. View the Phase and Status of your request.

5.5.3 Test Price List Setup

5.5.3.1 Setup Line Style

Follow this procedure for setting up a priced line style.

Steps

1. From the Contracts Manager Responsibility, navigate to Setup > Contract > Categories and Sources > Define Line Style.

2. If you create a new Line Style, enter a name for the Line Style for Line Type "FREE FORMAT". You can also use an existing line style.
3. Select the Priced check box.
4. Close the form and save.

5.5.3.2 Add Line Style to Category

Use this procedure to map your line style to a category.

Steps

1. Navigate to Categories and Sources > Define Categories.
2. Select a Category Code.
3. Navigate to Line Styles.
4. Navigate to File > New.
5. Select Top Line Style.
6. Close the form and save your changes.

5.5.3.3 Author the Contract Line

This procedure is followed when authoring a contract using the Line Style you just added.

Steps

1. Navigate to Launch Contracts > Open.
2. Navigate to Tools > New.
3. Select the Contract Category.
4. Click Create.
5. Enter the header information.
6. Select Line Items.
7. Select the Line Style you created.
8. Select the Name of the product you entered in the price list.
9. Enter quantity.
10. Select UOM. The price you entered should be returned to the line.

5.6 Set Up Pricing for Customer Products

Other than inventory based items, a user can price any item, using Oracle Pricing in Oracle Contracts following these setup steps. For this scenario, items to be priced will come from OKX_CUSTOMER_PRODUCTS_V. The jtot_object_code for this view is OKX_CUSTPROD.

For this example, a source called OKX_CUSTOMER_PRODUCTS_V which is an external source holding customer items will be used. OKX_CUSTPROD is the jtot_object_code for this source.

5.6.1 Define a Value Set

OKX_CUST_VAL gets its values from OKX_CUSTOMER_PRODUCTS_V view. Please refer to *Oracle Applications Developer's Guide* for additional information for defining value sets.

Steps

1. From the Application Developer Responsibility, navigate to Application > Validation > Values > Values, Effective.
2. Enter name for Value Set.
3. Enter value.
4. Enter a description for value.
5. Select the Enabled check box.
6. Save your work.

5.6.2 Define Segment for Context Item

Follow these steps to define the Free Format segment for pricing context ITEM. For more information on defining flexfields see the *Oracle Applications Flexfield Guide.*

Steps

1. From the System Administration Responsibility, navigate to Application > FlexField > Descriptive > Segments.
2. Run the query for Oracle Pricing and Pricing Contexts for Application and Title.
3. Select the ITEM in the Context Field Values region.
4. Click Segments. The Segments Summary window opens.

5. From the Segments Summary window, click New. The Segments window opens.
6. From the Segments window, enter "OKX_CUST_PROD".
7. Enter Description (Optional).
8. Select Pricing AttributeXX for Column.
9. Enter a number. This can be sequential with the other Segments.
10. Click Value Set.
11. Select "char_only".
12. Click Value Sets.
13. Close Value Sets window. You should see your new segment in the Segments Summary window.

5.6.3 Map Item Attribute

Steps

1. From the Pricing Manager Responsibility, navigate to Setup > Attribute Mapping.
2. Run a query for Line Pricing Attrs.
3. Select PRICING_ATTRIBUTEXX.
4. Click Defaulting Rules. The Attribute Defaulting Rules window opens.

5.6.4 Defining Function Returning Attribute Value

Follow these steps for mapping the pricing segment that you just defined, Pricing_attributeXX, to ojtot_object_code OKX_CUSTPROD. This result would be, if Oracle Pricing finds a linestyle with this source in the populated global table, then it treats the value of this source as the value for Pricing_attributeXX (OKX_CUST_PROD) for pricing context 'ITEM'.

For this, you are using a predefined OKC function defined in package OKC_PRICE_PUB that takes in the name of the global table which will hold the value at runtime and the OKX source against which the value will be held. The global table G_LSE_TABLE in this case is predefined and is being already used for seeded mappings. This mapping is exactly similar to other seeded mappings for line styles except the

source has been changed to 'OKX_CUSTPROD' - the jtot_object_code for view 'OKX_CUSTOMER_PRODUCTS_V'.

Steps

1. From the Defaulting Conditions region, enter "1" as Precedence.
2. Select "OKC Line Item Context" for Defaulting Condition.
3. Select the Enabled check box.
4. From the Default Sourcing Rules region, enter "1" for Sequence.
5. Select PL/SQL API as the Source Type. The Defaulting Rules Flexfield window appears.
6. Enter "OKC_PRICE_PUB" as the Package Name.
7. Enter "Get_LSE_SOURCE_VALUE(OKC_PRICE_PUB.G_LSE_TBL, 'OKX_CUSTPROD')" for Function Name.
8. Click OK.

5.6.5 Define Customer Product in Pricelist

Following this procedure is mandatory before rebuilding sourcing rules as the new version of the concurrent program. The reason is, building sourcing rules takes only those mapping into consideration that are being used somewhere either in modifier or Pricelists. This improves the performance of Pricing.

Steps

1. From the Oracle Pricing Manager Responsibility, navigate to Price Lists > Price List Setup > List Lines.
2. Run a query for price list or create a new price list.
3. Select "Item" for Product Context.
4. Select "OKX_CUST_PROD" for Product Attribute.
5. Select a Product Value from the LOV.
6. Select a UOM.
7. Select the Primary UOM check box.
8. Enter a Value.
9. Save your work.

5.6.6 Build Sourcing Rules

Follow this procedure for submitting a concurrent request to run "Build Sourcing Rules".

Steps

1. From the Oracle Pricing Manager Responsibility, navigate to View Concurrent Requests.
2. Click the Submit a New Request button.
3. Select "Build Sourcing Rules".
4. Click Submit.
5. Run a query for your Request ID.
6. View the Phase and Status of your request.

5.6.7 Test Customer Item Setup

5.6.7.1 Set up Line Style

Follow this procedure for setting up a priced line style.

Steps

1. From the Contracts Manager Responsibility, navigate to Setup > Contract > Categories and Sources > Define Line Style.
2. If you create a new Line Style, enter a name for Line Style for Line Type "CUST_ITEM" with source OKX_CUSTPROD. The UI name for this source is Customer Product. You can also use an existing line style.
3. Select the Priced check box on this line itself or any of its sub line styles.
4. Close the form and save.

5.6.7.2 Add Line Style to Category

Use this procedure to map your line style to a Category.

Steps

1. Navigate to Categories and Sources > Define Categories.
2. Select Category Code.

3. Navigate to Line Styles.
4. Navigate to File > New.
5. Select Top Line Style (CUST_ITEM).
6. Close the form and save changes.

5.6.7.3 Author the Contract Line

Follow this procedure when authoring a contract using the Line Style you just set up.

Steps

1. Navigate to Launch Contracts > Open.
2. Navigate to Tools > New.
3. Select Contract Category.
4. Click Create.
5. Enter the header information.
6. Select Line Items.
7. Select the Line Style you created.
8. Select the Name of the product you entered in the price list.
9. Enter the quantity.
10. Select UOM. The price you entered should be returned to the line.

5.7 Set up Pricing for a Determinant Component

Through Oracle Contracts and Oracle pricing, the user can use most of the components defined in a contract to be a determinant in pricing the item. The components includes linestyle items, rules and party roles.

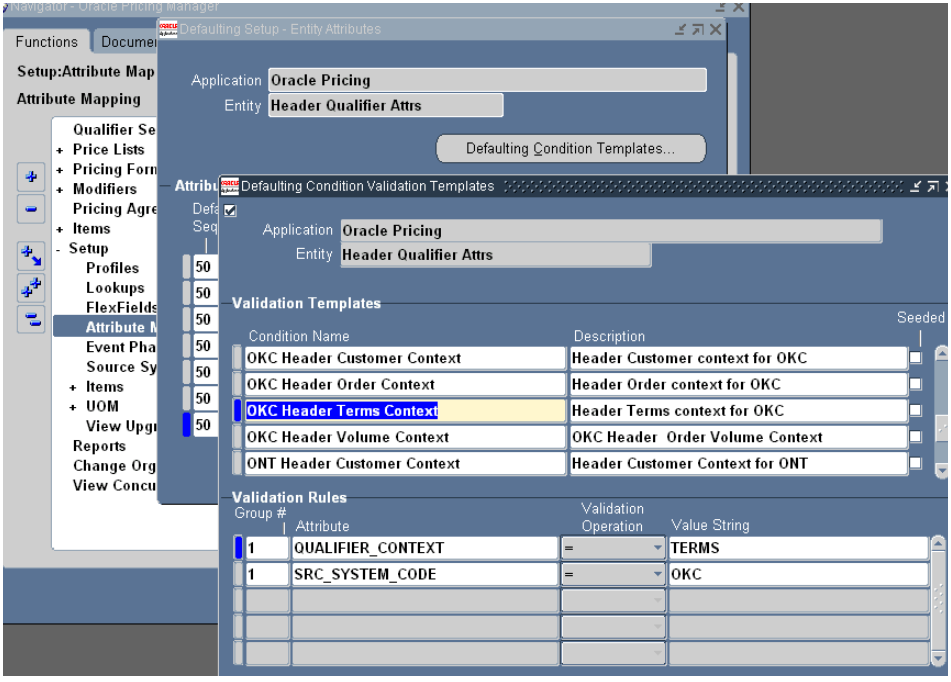
This can be done by mapping the component that needs to be used to some Qualifier Context Attribute or Pricing Context Attribute, depending on the way you are planning to use it. For example, payment terms affect the price for an item. Perhaps you would like to give special discounts or apply some surcharge on unit price based on the payment terms. For this, you define discounts that have payment term's value as a qualifier. The value for this qualifier should be picked from the payment rule defined in the contract.

You need to have a contract where there is no discount applied yet. Select a line, that can be priced again after the new discount comes into picture.

5.7.1 Map the Qualifier

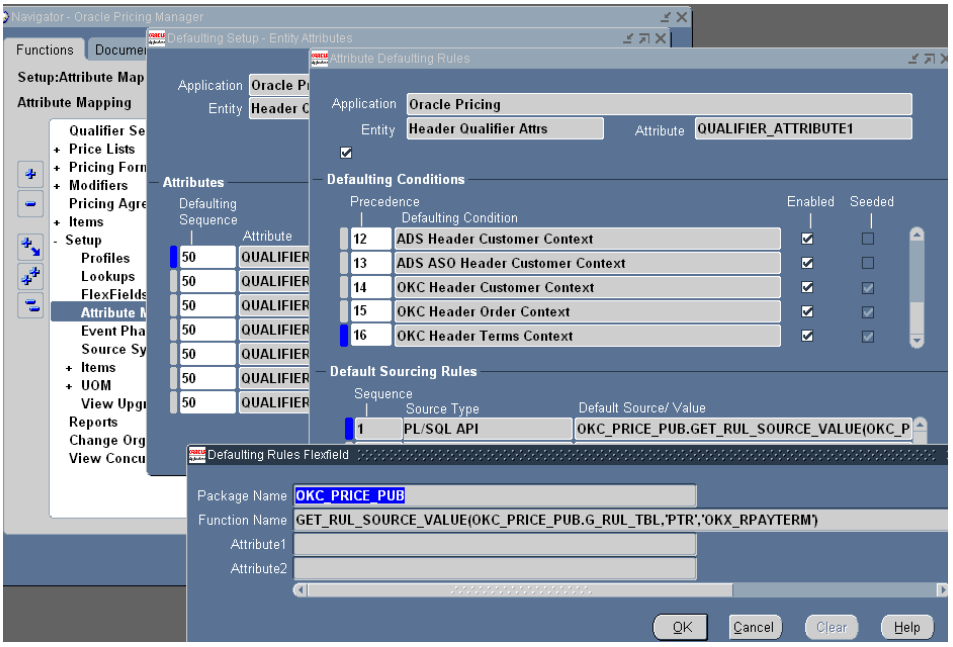
Steps

1. From the Pricing Manager Responsibility, navigate to Setup > Attribute Mapping.
2. Run a query for Header Pricing Attrs.
3. Select QUALIFIER_ATTRIBUTEXX.
4. Click Defaulting Condition Template for Qualifier TERMS with source system OKC. The Defaulting Condition Validation Templates window opens. The condition shown is already seeded.
5. Close the Defaulting Condition Validation Templates window.



6. Click Defaulting Rules.

7. Select the seeded condition "OKC Header Terms Context.
8. Add the function which will provide value for this source.



Guidelines

In Oracle Pricing if you find a rule with this source 'OKX_RPAYTERM' in the populated global table for rules, then treat the value of this source as the value for Qualifier_attribute1 for qualifier context "TERMS". For this predefined Contracts Core function defined in package OKC_PRICE_PUB that takes in the name of the global table which will hold the value at runtime and the OKX source against which the value will be held. The global table G_RUL_TBL in this case is predefined and is used for seeded mappings. The source here "OKX_RPAYTERM" is the jtot_object_ code for the rule source for rule "PT". The value for the rule is picked from value defined for this rule in rule group "PAYMENT". Again, this is a seeded mapping.

Note: In Oracle Pricing, the Qualifier mapping can be done at header or line level. Shown here with header level mapping. The mapping done at header level is used to qualify modifier/price list headers. The mappings done at line level are used to qualify list lines. In this example, the list header is qualified.

5.7.2 Define the Discount

Steps

1. From the Pricing Manager Responsibility, navigate to Modifiers > Modifier Setup.
2. Define a discount at list line level.
3. Click List Qualifiers.
4. Enter the qualifier that you just defined with the value that you intend to use in the contract.
5. Close the window.

5.7.3 Build Sourcing Rules

Run the Concurrent program - 'Build Sourcing Rule'. Application- "Oracle Pricing".

5.7.4 Testing

Steps

1. From the Contracts Manager responsibility, navigate to Launch Contracts Open > View > Find.
2. Enter the contract number.
3. Click search.
4. Open the contract.
5. Click Rules.
6. Select the Payment for Rule Group (where the qualifier can be picked up).
7. Reprice the contract line. The defined discount with the qualifier should be returned.

5.8 Troubleshooting Pricing Integration

The following are questions and answers to common problems:

- How do I switch to Advanced Pricing?

  If you have old contracts that were using Pricing rule, run the Upgrade concurrent program. This program will populate the new fields and enable OKC: Enable Advanced Pricing profile option.

- Why don't I see the Adjustments tab in Contracts Authoring?

  Enable the OKC: Enable Advanced Pricing profile option. If the intent of the Contract is BUY, the Adjustments tab will not appear.

- I entered the pricelist with item and have entered UOM and Qty. Why am I getting the 'Invalid Pricelist/item Combination…' error?

  If you were trying to price an inventory item, then in the Linestyle form verify that the linestyle for the item that you are pricing has the 'Item_to_price_yn' flag checked.

  If you were trying to price a customer item or a free format line, validate your steps against those mentioned in: *Setting Up Pricing with Contracts Core.*

  If everything is fine, recheck the UOM and item in Pricelist and authoring form.

- I did not specify the pricelist as I want to get the best list price but I get the 'No Pricelist…' error. Why do I get this error message?

  Check that the item that you are trying to price is in a some Pricelist.

  Check that the User search flag is set to 'Yes' in phase 'Line List Price'. To check follow these steps:

  - Switch to the Oracle Pricing Manager Responsibility.
  - Navigate to Setup > Event phases.
  - Perform Query "All".
  - In the 'Event phases' block of 'Batch' pricing event, either seeded search flag or user search flag should be 'Yes'

- When pricing is called, why do I get an "Unexpected" error message?

  If you get "Unexpected Pricing Error in procedure Build_Context for Header. SQL….", then the problem is in some attribute mapping. Rerun the Build Sourcing Rules concurrent program and check the output details.

If you get "Unexpected Pricing Error in procedure price_request" error, some QP packages might be invalid. Recompile invalid packages.

- How does Advanced Pricing behave for recursive line styles?

  Any Linestyle can be recursive in the current setup. Following are the various possible scenarios:

  - When the linestyle holding the item (Inventory or Customer item) is recursive.
  - When the linestyle holding Priced flag is recursive.
  - When the free format linestyle is recursive as well.

  In the above scenarios, the basic rule that it is followed is that if there are lines and sub lines with same flags, then only one instance really holds the meaning. The exception to this rule is when a linestyle is used by configurator. For configured items, all the line styles in the configuration are considered Priced and holding item to price irrespective of the fact they are top line or sublines of themselves.

  For all the non configurator lines, following apply:

  - If the priced linestyle is recursive, only the first occurrence of priced line style from top to bottom will hold its Priced meaning.
  - If the item_to_price_yn flagged line style or Customer item line style is recursive, the one closest to Priced line style, among its parents will hold the meaning. It could be possible that a line style is recursive, priced and item to price. In that case the first occurrence (rule above) will hold the meaning.
  - If the free format line is recursive, the one closest to the Priced line style, among its parents will hold the meaning.

- Why can’t I override the net price?

  Check that the 'OKC: Manual adjustment allowed' profile option is enabled

  There should be a qualifying override, manual adjustment available. To check this, click the Adjustments LOV in the Adjustments tab. If there are any records there with over ride flag checked, that means it is available.

6

Integrating Contracts Core with Oracle Configurator

Oracle Contracts Core/Sales Contracts has the Contract Authoring form which is used to enter contract information. This form has a lines block to enter lines information. Configurations are entered from the Lines tab. This chapter covers the following topics as they relate to the setup of the integration to Oracle Configurator:

- Integration with Oracle Configurator
- Properties of Configured Items in Contracts
- Related Setups

References:

Oracle Configurator Installation Guide

Oracle Configurator Implementation Guide

Oracle Configurator User's Guide

Oracle Configurator Release Notes

Oracle Pricing User's Guide

Oracle Contracts Core User Guide

6.1 Integration with Oracle Configurator

You can create configured items from model items while creating contracts. You can view the selected items that make up the configured item. The configured item can be priced. A recursive and priced line style is selected along with the model item.

The Contracts Authoring form recognizes this combination and enables "Configure Item" from the Action menu.

6.2 Properties of Configured Items in Contracts

A line style is used in conjunction with other available line styles to create a line structure. No modifications are allowed to child lines, modifications will be allowed only in the top line (usually model used for configuration). In the top line, changes are allowed to Rules (if rules are changed, user will be asked if contract should be repriced) and Pricing (manual modifiers and pricing over-ride functionality will be available in top line only). From the Adjustments tab, no changes are allowed for the child lines.

All contract rules (i.e. ship to, or bill to) are assigned to the top-line of the recursive line only. This is consistent with the fact that in a recursive line style, only the top line is available at time of creating the configuration. No changes are allowed to a configured item, for an active contract.

6.3 Related Setups

The following topics are related to setup:

- Profiles Setup
- Port Setup
- Configurator Database Setup
- Process Specific Setup

6.3.1 Profiles Setup

These are setup using the Oracle Applications, System Administrator responsibility:

Profile	Settings
OKC: Enable Advance Pricing	Yes (At site or application level). Also, the Adjustments tab, is enabled only for Intent of "Sell".
BOM: Configurator URL of UI Manager	This is the only URL that is needed to invoke configurator. URL example: http://<ApacheServerName>.domain:<port>/oa_servlets/oracle.apps.cz.servlet.UiServlet

6.3.2 Port Setup

The following table displays an example of a Port Setup:

Item	Definition
Product	okc
Port#	Your port number
Hostname	Machine name
Apache Home	Home path
Appl Top	Directory path
Prepend LD Library Path	(=> $CZ_TOP/bin)
DBC Filename	Your database env dbc file name. This file is located in $FND_TOP/secure
Database Name	Your database name
Apps Password For DAD	apps****
Servlet Classpath Repositories	The entries specified in this field will be used to populate repositories clause in the zone.properities file. Servlets/classes specified in this location will be reloaded on change.
Prepend System Classpath	Same as Servlet Classpath Repositories
JSP Classpath	Same as Servlet Classpath Repositories
Oracle Home	/oracle/8.0.6
Character Set	Non UTF

Item	Definition
Java Runtime Variables	Following are the four variables for the CZ invocation (Depending on the CZ port setup location): ▪ cz.activemodel=/lp|/dp|/atp| ▪ cz.uiserver.ciolog=/export/home/dbadmin/db/ias1.3.9_1.2.2p/Apache/Apache/ok_pf_5800/logs/ciolog.txt ▪ cz.uiservlet.formtemplateurl=http://<ApacheServerName>.domain:<port>/OA_HTML/US/czFormTemplate.htm. ▪ cz.uiservlet.blaftemplateurl=http://<ApacheServerName>.domain:<port>/OA_HTML/US/czFormTemplate.htm.
InitArgs Variables & Zone	NULL

6.3.3 Configurator Database Setup

For ERROR: 1

Batch Validate User is not provided in dbc file java.lang. Error: Cannot start session for the following reason:

Solution:

The following two lines are needed in the.dbc file:

BATCH_VALIDATE_USER=sysadmin

BATCH_VALIDATE_FND=sysadmin

For ERROR: 2

Message: Xinit java.lang.UnsatisfiedLinkError: Xinit

Solution:

1. Make sure you set LD_LIBRARY_PATH in Quik Apache port set-up to $CZ_TOP/bin.
2. Navigate to $APPL_TOP there will be CZ directory.
3. Check for $CZ_TOP/bin/libczlce.so file exists. Required by CZ.

   cd $CZ_TOP

```
cd bin
ldd libczlce.so       (will display the following files)
      libclntsh.so.1.0 => /oracle/8061f608121/lib/libclntsh.so.1.0
      libnsl.so.1     => /usr/lib/libnsl.so.1
      libsocket.so.1   => /usr/lib/libsocket.so.1
      libdl.so.1      => /usr/lib/libdl.so.1
      libc.so.1       => /usr/lib/libc.so.1
      libaio.so.1     => /usr/lib/libaio.so.1
      libm.so.1       => /usr/lib/libm.so.1
      libmp.so.2      => /usr/lib/libmp.so.2
      /usr/platform/SUNW,Ultra-Enterprise/lib/libc_psr.so.1
```

4. In case you get file not found, you need to copy the file.
5. After setting your environment, relink this file with following at command line.

   adrelink force=y "cz libczlce.so"

6.3.4 Process Specific Setup

Prerequisites

The CZ Patch#1844469 has to be successfully applied on your database.

Steps

1. Prepend LD Library Path: /afrz/crmcon/kdv1155a/cz/11.5.0/bin. The quick-apache-server's prepend load library path should point to the directory that contains the file libczlce.so ($CZ_TOP/bin).
2. Prepend System Classpath: /afrz/crmcon/kdv1155a/java/apps.zip.
3. Ensure the four Java Runtime Variables exist. Mainly, For Pricing calls: cz.activemodel=/lp|/dp|/atp| is required.
4. Re-Register your Quick Apache port and re-start for above changes to take effect.
5. Ensure that you see the Configurator version by typing in the BOM URL with a suffix (e.g. ?test=version).

6. If the database is refreshed, ensure the Configurator related profiles exist. Refer to the CZ logs for any errors, and possible fixes listed above.

7. Ensure the linestyle used for configuring model items is defined to be: Recursive, Item to Price, and Price check boxes are selected. The source should be any OKX object which points to MTL_SYSTEM_ITEMS. OKX_SYSTEM_ITEMS_V, e.g. NonService Item linestyle is used. Also, ensure that the new linestyle is attached to the category (subclass) that will be used to configure any model via Oracle Contracts Core.

8. If a user wishes to enable automatic pricelist search, in Advanced Pricing mode, then the following setup needs to be done in Oracle Pricing (QP) Events Phases window. With this setup, the Pricing Engine will pickup the lowest precedence pricelist in case the user does not select any pricelist prior to invoking configurator. The QP seeds the search flag as "No" and user can override this to "Yes ". This is set in the Oracle Pricing responsibility.

A

Contracts Core Profile Options

This appendix displays a table showing Oracle Contracts Core Profile Options.

A.1 Setting up Profile Options

Use this list to identify the profile options you need for your implementation. You can set these profile options in any order you like. To change profile options, refer to the *Oracle Applications Users Guide.*

Profile	Description	Level
OKC: Schedule Rule Alert Window	The number of days before a due task, that the user is notified of upcoming task.	Responsibility and User
OKC: Change Request Approval	This overrides the workflow approver.	Any level
OKC: Contract Approver	This overrides the workflow approver.	Any level
OKC: Schedule Rule Escalate	Number of days after a task has missed its due date when escalation begins.	Responsibility and User
OKC: Status Change Batch Size	Determines the number of records to be updated before they are saved in the database (this parameter should be fine-tuned by the database administrator).	Site and Responsibility
OKC: Renewed Contract Identifier	This identifier will be attached to the renewed contract's contract number.	Any level
OKC: Public Group Creator	Privilege to create a public group in Oracle Contracts.	Responsibility and User

Profile	Description	Level
OKC: Time UOM Class	Limits the units you see in the Map Time Units window.	Site
OKC: Update Negotiated Price	This drives the default of the Update Negotiated Price ON/OFF feature found within the Action menu.	Responsibility and User
OKC: Default Group for Contracts Created from Quote	This profile should be set to a contract group. Contracts created from Oracle *i*Store or Oracle Quoting will automatically be organized in this contract group found in the Contract Navigator. This takes precedence over any groups set up in a template used to create a contract. The public contract group must first be created before specifying that group name.	Any level
OKC: Success Message Recipient	This enables a recipient to be notified regarding the success of an outcome. This value can be overwritten in case the user selects another person to be notified when entering the outcome condition.	Site
OKC: Error Message Recipient	This is the resource (person) who will be notified in case the outcome execution has failed. This value can be overwritten in case the user selects another person to be notified when entering the outcome condition.	Site
OKC: Event Listener Iterations	Allows System Administrators to set the work load of the listeners. The default load (listener its/iterations) is set to 100.	Site and Application
OKC: Outcome Listener Iterations	Allows System Administrators to set the work load of the listeners. The default load (listener its/iterations) is set to 100.	Site and Application
OKC: Number of days to retain a message on the exception queue	In case of any exceptions encountered while running events, messages are thrown into the exception queues. This profile determines the retention period (in days) of such messages.	Site
OKC: After Report Procedure	Use this profile if you want to perform a validation after the contract is printed.	Any level
OKC: Allow Manual Price Adjustment	This profile enables users to manually adjust the final price of a line or header. The default value is "Yes". This profile is valid only when users enable Advanced Pricing.	User, Responsibility, or Application

Profile	Description	Level
OKC: Contract template for standard terms and conditions	When an Oracle iStore customer checks out, the terms and conditions that are viewed and the subsequent contract that is created for that iStore order, are originated from the contract template that is defined in this profile option. Choose from the list of contract templates.	Site, Application, and Responsibility
OKC: Default Opportunity Campaign Code	This profile option is used with Opportunity integration. When an opportunity representing a contract is created, it must be associated with a marketing campaign code. Set this profile to the campaign code that should be defaulted for each new opportunity created from Contracts.	Any level
OKC: Default contract administrator for notifications	This profile should be set to the person who will receive notifications that a contract in "Entered" status was created from Oracle iStore. This will occur when the customer disagrees to the standard terms and conditions thereby initiating creation of a new contract and notifying the default contract administer identified in this profile option.	Any level
OKC: Notify administrator about new contract from iStore	Set this profile option to "Yes" if you want the contract administrator to receive notifications for contracts created from Oracle iStore when the customer disagrees with standard terms and conditions. If set to "No", the contract administrator will not be notified when contracts are created from Oracle iStore.	Any level
OKC: Document Sequence Name	This is the sequence name that will be used to number contracts. The sequence is automatically created in Application Developer by Contracts.	Site, Application, and Responsibility
OKC: Enable Advanced Pricing	Use this profile to indicate whether the users want to use Oracle Pricing for price related calculations or not.	Site or Application
OKC: Report prerun validation procedure	Use this profile if you want to perform a validation before the contract is printed.	Any level
OKC: Set environment procedure	Set contract specific context in a new session.	Any level

Profile	Description	Level
OKC: Suppress Sending Emails from Contract Alert Workflow	Use this profile to prevent or allow emails to be sent from Oracle Workflow after Contract Alert has executed.	Any level
OKC: User Directory	Used for *i*Store integration. This profile provides the location of the style sheet that based on it, the terms and conditions for Oracle License Agreement are formatted.	Site
OKC: Discoverer Reports Enabled	Use this profile to enable site specific Discoverer reports feature. Default is set to "No".	Site
OKC: Alert Time-out in Minutes	This profile determines the time-out period (in minutes) for failed outcomes, after which Oracle Workflow will retry automatically to execute the outcome. Within this period, the user can manually go and retry or cancel the process from the "Inbox". This value comes into effect only if the user has not responded within the time-out period. The default is two days (2880 minutes).	Site
OKC: Contracts Expired Past Days	Use this profile to allow users to set the value of the number of days past since a contract has expired. For example, contracts expired for the past 60 days, the profile option will contain the value 60. This value is utilized in the search criteria.	Any level
OKC: Global Update Privileges	The Update Online function allows users to update "Active" and "Signed" type contracts online. To enable the Update Online function: ■ Set the OKC: Global Update Privileges profile option to "Yes" ■ Grant the Update Online function to the user responsibility	All

B

Contracts Core Lookup Codes

This appendix displays a table showing Oracle Contracts Core Lookup Codes.

B.1 Setting up Lookups

Use this table to identify the lookup codes (Quick Codes) that you need to define for your implementation. Use the Application Object Library Lookups window to identify and define the Lookup Codes.

Code	Description	Level
OKC_ARTICLE_SET	Defines sets of articles.	User
OKC_BILLING_RATE_CODE	Identifies Billing Rate Code.	System
OKC_CHANGE_REQUEST_STATUS	Identifies change request status.	System
OKC_CONTACT_ROLE	Defines party contact roles.	Extensible
OKC_CONTINGENT_EVENTS	Defines contingent events for execution.	Extensible
OKC_CONTRACT_SOURCES	Identifier of source system of a contract.	Extensible
OKC_INHERITANCE_TYPE_RDF	Identifies the Inheritance type.	System
OKC_ISTORE_ART_TITLE	Identifies the contract articles page title.	System
OKC_LINE_RENEWAL_TYPE	Identifies types of renewals for contract lines.	System

Code	Description	Level
OKC_LINE_TYPE	Defines line types.	Extensible
OKC_PRICE_TYPE	Defines price types.	Extensible
OKC_PROCESS_ USAGE_TYPES	Defines process usage types.	Extensible
OKC_QA_ERROR_ LEVEL	Identifies details for QA report printing.	System
OKC_RELATED_OBJ	Shows contract related objects.	System
OKC_RENEWAL_ PRICING_TYPE	Shows types of pricing for renewed contract.	System
OKC_RENEWAL_TYPE	Shows types of renewals for contracts.	System
OKC_ROLE	Defines party roles for a contract.	Extensible
OKC_CONTACT_ROLE	Defines contact roles for a contract.	Extensible
OKC_RULE_DEF	Identifies Rule definitions setup.	System
OKC_RULE_GROUP_ DEF	Place holder for all rules to be assigned to a contract.	Extensible
OKC_SEED_ACCESS_ LEVEL_SEU	Used for identifying protection level of a record -SEU.	System
OKC_SEED_ACCESS_ LEVEL_SU	Used for defining protection level of a record -SU.	Extensible
OKC_STATUS_TYPE	Identifies Status Types.	System
OKC_SUBJECT	Defines standard article subjects.	User
OKC_STS_CHG_ REASON	Used for defining status change reason	Extensible