ORACLE
PEOPLESOFT ENTERPRISE

# PeopleSoft Enterprise Components for FSCM 9.0 PeopleBook

**September 2006**

ORACLE

PeopleSoft Enterprise Components for FSCM 9.0 PeopleBook
SKU FSCM9ECP-B 0806
Copyright © 1992-2006, Oracle. All rights reserved.

The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

If the Programs are delivered to the United States Government or anyone licensing or using the Programs on behalf of the United States Government, the following notice is applicable:

**U.S. GOVERNMENT RIGHTS**

Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software--Restricted Rights (June 1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.

The Programs may provide links to Web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party Web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

Oracle, JD Edwards, PeopleSoft, and Siebel are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

**Open Source Disclosure**

Oracle takes no responsibility for its use or distribution of any open source or shareware software or documentation and disclaims any and all liability or damages resulting from use of said software or documentation. The following open source software may be used in Oracle's PeopleSoft products and the following disclaimers are provided.

*Apache Software Foundation*

This product includes software developed by the Apache Software Foundation (http://www.apache.org/). Copyright © 2000-2003. The Apache Software Foundation. All rights reserved. Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0.

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

*OpenSSL*

Copyright © 1998-2005 The OpenSSL Project. All rights reserved.

This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/).

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

*Loki Library*

Copyright © 2001 by Andrei Alexandrescu. This code accompanies the book: Alexandrescu, Andrei. "Modern C++ Design: Generic Programming and Design Patterns Applied". Copyright © 2001 Addison-Wesley. Permission to use, copy, modify, distribute and sell this software for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

*Helma Project*

Copyright © 1999-2004 Helma Project. All rights reserved. THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE HELMA PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Helma includes third party software released under different specific license terms. See the licenses directory in the Helma distribution for a list of these license.

*Sarissa*

Copyright © 2004 Manos Batsis.

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA.

*ICU*

ICU License - ICU 1.8.1 and later COPYRIGHT AND PERMISSION NOTICE Copyright © 1995-2003 International Business Machines Corporation and others. All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, provided that the above copyright notice(s) and this permission notice appear in all copies of the Software and that both the above copyright notice(s) and this permission notice appear in supporting documentation. THE SOFTWARE IS PROVIDED "AS IS," WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE. Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

All trademarks and registered trademarks mentioned herein are the property of their respective owners.

*Sun's JAXB Implementation – JDSDK 1.5 relaxngDatatype.jar 1.0 License*

Copyright © 2001, Thai Open Source Software Center Ltd, Sun Microsystems. All rights reserved.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

*W3C IPR SOFTWARE NOTICE*

Copyright © 2000 World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved.

Note: The original version of the W3C Software Copyright Notice and License could be found at http://www.w3.org/Consortium/Legal/copyright-software-19980720.

THIS SOFTWARE AND DOCUMENTATION IS PROVIDED "AS IS," AND COPYRIGHT HOLDERS MAKE NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF THE SOFTWARE OR DOCUMENTATION WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS. COPYRIGHT HOLDERS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF ANY USE OF THE SOFTWARE OR DOCUMENTATION.

# Contents

**General Preface**

**About This PeopleBook Preface ..... xvii**

PeopleSoft Enterprise Application Prerequisites ..... xvii

Application Fundamentals ..... xvii

Documentation Updates and Printed Documentation ..... xviii

Obtaining Documentation Updates ..... xviii

Downloading and Ordering Printed Documentation ..... xviii

Additional Resources ..... xix

Typographical Conventions and Visual Cues ..... xx

Typographical Conventions ..... xx

Visual Cues ..... xxi

Country, Region, and Industry Identifiers ..... xxi

Currency Codes ..... xxii

Comments and Suggestions ..... xxii

Common Elements Used in PeopleBooks ..... xxii

**Preface**

**PeopleSoft Enterprise Components Preface ..... xxv**

Overview of PeopleSoft Enterprise Components ..... xxv

## Part 1
## Getting Started With PeopleSoft Enterprise Components

**Chapter 1**

**Getting Started With PeopleSoft Enterprise Components ..... 3**

Getting Started With PeopleSoft Directory Interface ..... 3

Getting Started With Common Objects and Components ..... 4

Getting Started With Enterprise Integration ..... 4

## Part 2
## PeopleSoft Directory Interface

Copyright © 1992-2006, Oracle. All rights reserved.

**Chapter 2**
**Using PeopleSoft Directory Interface.....7**
Understanding PeopleSoft Directory Interface.....7
Overview of Using PeopleSoft Directory Interface.....8
Setting Up in PeopleSoft Application Designer and PeopleSoft Integration Broker.....9
Using the Directory Configurations Component.....9
Common Elements Used in This Chapter.....10
Defining and Configuring the Directory.....10
Pages Used to Define and Configure the Directory.....11
Configuring the Directory Connection.....11
Caching the Schema.....13
Deleting the Directory Configuration.....14
(Optional) Setting Up Directory Authentication.....14
Pages Used to Set Up Directory Authentication.....15
Using Map Authentication.....15
Viewing User Properties.....16
Setting Up Mappings.....17
Understanding Mappings.....18
Pages Used to Set Up Mappings.....18
Creating Mappings.....18
Modifying the Distinguished Name.....20
Specifying Distinguished Name Details.....21
Translating or Performing Functions With Database Values.....22
Specifying Distinguished Name Defaults.....23
Mapping Data to Directory Object Class Attributes.....24
Locating Delivered Messages.....26
(Optional) Setting Up Entry Membership Rules.....26
Pages Used to Set Up Entry Membership Rules.....26
Creating Entry Definitions.....26
Specifying Entry Membership Rules.....28
Loading Data into the Directory.....30
Understanding Directory Load Behavior.....30
Pages Used to Load PeopleSoft Data into the Directory.....30
Loading the Directory with PeopleSoft Data.....30

**Chapter 3**
**Reviewing Directory Data and Generating Reports.....33**
Reviewing LDAP Directory Data.....33
Pages Used to Review Directory Data.....33


Copyright © 1992-2006, Oracle. All rights reserved.

Running a Directory Audit....33
Running a Directory Search....34
Viewing PeopleSoft Directory Interface Reports....35
Pages Used to View PeopleSoft Directory Interface Reports....36
Viewing the Directory Audit Report....36
Viewing the Business Interlink Status Report....36
Managing Transaction Audit History....37
Pages Used to Manage Transaction Audit History....38
Running a Transaction History Report....38
Searching for a Transaction History....39
Purging Transaction History....40

# Part 3
# Common Objects and Components

**Chapter 4**
**Working With Currencies and Market Rates....43**
Understanding Currencies and Market Rates....43
Understanding Currency and Market Rate Tables....43
Understanding Triangulation....44
Understanding Conversion Factor Fields and the Visual Rate....45
Understanding Application-Specific Requirements for Currency Conversion....46
Defining Currencies....46
Page Used to Define Currencies....46
Defining Currency Codes....47
Defining Currency Quotation Methods....48
Page Used to Define Currency Quotation Methods....48
Defining Currency Quotation Methods....48
Defining Market Rates....50
Pages Used to Define Market Rates....51
Defining Market Rate Indexes....51
Defining Rate Types....52
Creating Market Rate Definitions....53
Defining Market Rates....55
Accessing Exchange Rate Details....57
Calculating Cross, Triangulated, and Reciprocal Rates....59
Understanding the EOP_RATECALC process....59
Page Used to Calculate Cross, Triangulated, and Reciprocal Rates....59

Copyright © 1992-2006, Oracle. All rights reserved.

Running the EOP_RATECALC Process....59
Using the Currency Exchange Calculator....61
Page Used to Use the Currency Exchange Calculator....62
Converting Amounts Using the Currency Exchange Calculator....62

**Chapter 5**
**Using Datasets....63**
Understanding Datasets....63
Defining Dataset Rules....63
Understanding Dataset Rules....63
Page Used to Define Dataset Rules....64
Creating Dataset Rules....64
Defining Dataset Roles....65
Page Used to Define Dataset Roles....65
Defining Dataset Roles....65
Defining Mobile Data Distribution....65
Understanding Mobile Data Distribution....66
Pages Used to Define Mobile Data Distribution....66
Defining Mobile Data Distribution Rules....66
Using Mobile User Rules....68

**Chapter 6**
**Using Interactive Reports....69**
Understanding Interactive Reports....69
Common Elements Used in This Chapter....69
Configuring Interactive Reports....70
Pages Used to Configure Interactive Reports....70
Defining Reports....70
Defining Servers....71
Defining ODBC Connections....72
Configuring the Environment....73
Mapping Queries....73
Viewing Query Prompts....74
Validating the Environment....75

**Chapter 7**
**Setting Up the Credit Card Interface....77**


Copyright © 1992-2006, Oracle. All rights reserved.

Understanding Credit Card Processing.....77
Understanding a CyberSource Integration.....77
Understanding an XML-Based Integration.....78
Understanding Manual Processing.....78
Setting Up the Credit Card Interface.....79
Pages Used in Setting Up the Credit Card Interface.....79
Defining Connection Parameters.....79
Defining Accepted Credit Card Types.....82
Testing the Credit Card Interface.....83
Testing Credit Card Transactions.....85
Configuring the Java Interlink Plug-In.....88
Understanding the Java Interlink Plug-In.....88
Setting Up the CyberSource API and Java Plug-in on Microsoft Windows NT Systems.....89
Setting Up the CyberSource API and the Java Plug-in on Solaris Systems.....90
Configuring the PeopleSoft Application Designer Plug-in.....92
Configuring Access to Account Information.....93
Configuring Proxy Server Support.....96
Setting Up Tracing.....96
Setting Parameter Check Logic.....97
Setting the Decryption Option.....97
Setting Transaction Inputs and Outputs.....97
Identifying Required Parameters.....98
Using Authorize-and-Bill Transactions.....98
Using Authorize-Only Transactions.....99
Using Bill-Only Transactions.....100
Using Credit Transactions.....100
Using the Address Verification Service.....102
Using the Fraud Screen Service.....103
Identifying Input Fields That Are Not Used or Supported.....104

# Part 4
# PeopleSoft Active Analytics Framework

**Chapter 8**
**Understanding PeopleSoft Active Analytics Framework.....107**
Understanding PeopleSoft Active Analytics Framework.....107
Understanding Policies and Trigger Points.....107
Understanding the Data Library.....108

Copyright © 1992-2006, Oracle. All rights reserved.

Understanding the Action Framework.....109

**Chapter 9**
**Building and Managing Policies.....111**
Building Policies.....111
Prerequisites to Building Policies.....111
Pages Used in This Section.....112
Building, Editing and Activating Policies.....112
Managing Trigger Points.....118
Removing a Policy or Policy Groups.....119
Reordering Policy or Policy Groups.....120
Adding a Policy or Policy Group.....120
Reusing Policies.....120
Adding a Precondition.....120
Setting Execution Options.....121
Understanding Execution Options.....121

**Chapter 10**
**Setting Up the Data Library.....127**
Understanding the Data Library.....127
Using Data Library Components.....127
Pages Used in This Chapter.....128
Creating Implementations.....128
Registering an Implementation.....129
Creating Terms.....130
Understanding Term Properties.....130
Using Generic Implementations.....132
Using Contextual Implementations.....132
Managing Terms.....132
Testing Term Implementations.....134

**Chapter 11**
**Managing Contexts.....137**
Understanding Contexts.....137
Configuring Contexts.....138
Pages Used in This Chapter.....138
Generating Context Objects.....138


Copyright © 1992-2006, Oracle. All rights reserved.

Managing Context Objects.....140

**Chapter 12**
**Setting Up the Action Framework.....143**
Understanding the Action Framework.....143
Architecture of the Action Framework.....143
Understanding Action Types.....143
Understanding How Actions Execute.....144
Registering Action Types and Action Type Bundles.....145
Pages Used in This Chapter.....145
Registering Action Types.....145
Registering Action Type Bundles.....147

**Chapter 13**
**Administering the Framework.....149**
Using the Setup Components.....149
Pages Used in This Chapter.....149
Setting Log and Installation Options.....150
Setting Data Library Log Options.....150
Installation Options.....151
Registering Action Objective.....153
Registering Trigger Types and Trigger Points.....154
Registering Trigger Types.....154
Registering Trigger Points.....155
Defining Subject Areas.....156
Registering Operators and Operator Sets.....158
Registering Operators.....158
Registering Operator Sets.....159
Registering Resolution Methods.....160
Registering Business Domain and Category.....161
Register Business Domain.....161
Register Category.....161

**Chapter 14**
**Considerations for Enabling the Framework.....163**
Considerations When Creating Contexts.....163
Considerations When Creating Custom Operator Expressions.....164

Copyright © 1992-2006, Oracle. All rights reserved.

Considerations When Creating a Trigger Point....165
Enabling a Trigger Point in a PeopleCode Event....166
Enabling the Display Action in a PeopleSoft Application Page....166
Considerations When Creating a New Action Type....166
Configuring the Action Type....166
Creating Design and Runtime Application Class Code....170
Considerations When Creating an Application Class Implementation....171
Technical Details of Application Class Implementations....171
Example of an Application Class Accessing Implementation Binds....172
Sample Methods of an Application Class Resolving Terms....172
Example of How a Value Can Be Passed to the Data Library....173
Sample Code of an Application Class Implementation....173
Considerations When Creating a PS Query Implementation....174
Considerations When Creating a Policy....175

# Part 5
# Enterprise Integration

**Chapter 15**
**Understanding Enterprise Integration....179**
Understanding PeopleSoft Business Interlinks....179
Understanding PeopleSoft Component Interfaces....179
Understanding File Layouts....179
Understanding PeopleSoft Integration Broker....180

**Chapter 16**
**Understanding Integration Points....181**
Overview of Integration Points....181

**Chapter 17**
**Activating Integration Points....183**
Setting Up PeopleSoft Integration Points....183
Pages Used to Set Up PeopleSoft Integration Points....183
Setting Up Publication Rules....184
Mapping Nodes to a Chunking Rule....185
Assigning Business Units or SetIDs to a Chunking Rule....185


Copyright © 1992-2006, Oracle. All rights reserved.

Specifying OnRoute PeopleCode.....185
Setting Up Related Languages.....186
Understanding Related Language Tables.....186
Understanding Related Language Guidelines for Messaging.....187
Interpreting Component Processor Behavior.....188
Publishing a Message from a Component.....188
Publishing a Message from Batch Programs.....189
Subscribing to Data in a PeopleSoft Multilingual Environment.....189
Subscribing to Data in a Non-Multilingual Environment.....189
Examining Related-Language Messaging Scenarios.....190
Publishing a Non-Base Language Message to a Subscribing System With a Different Base Language and No Prior Data.....191
Switching the Preferred Language to Japanese and Updating the Same Employee's Name and Address.....191

**Chapter 18**
**Assigning Publishing Rules.....193**
Understanding Publishing Rules.....193
The Publish Utility.....193
Terms Used When Defining Publish Rules.....193
Assigning Full Table Publishing Rules.....194
Pages Used to Assign Full Table Publishing Rules.....195
Associating a Rule to a Message.....195
Mapping a Message Record to Another Record.....196
Specifying Languages in Which to Publish Messages.....198
Assigning Batch Publishing Rules.....199
Pages Used to Assign Batch Publishing Rules.....199
Associating a Rule to a Message.....199
Mapping a Message Record to Another Record.....200
Assigning an Application Program to a Publishing Rule.....200
Setting Up Message Chunking.....201
Pages Used to Set Up Message Chunking.....202
Understanding Message Chunking.....202
Identifying When to Use Chunking.....203
Selecting Chunking Fields.....203
Creating Chunking Rules.....205
Defining the Chunking Rule Description.....206
Maintaining Chunking Data for Business Units.....206
Maintaining Chunking Data for SetIDs.....207
Adding Nodes to Existing Chunking Rules.....208

Copyright © 1992-2006, Oracle. All rights reserved.

Assigning Business Units or SetIDs to a Chunking Rule....208
Assigning Chunking Rules to a Business Unit....209
Assigning Chunking Rules to a SetID....210
Creating Custom Chunking Tables....210
Creating a Custom Chunking Table....211
Creating a View for the Component Search Record....211
Creating Maintenance Pages....211
Creating a Component....212
Creating Routing PeopleCode....213

**Chapter 19**
**Using the Error Handling Utility....215**
Understanding the Error Handling Utility....215
Error Management Process....215
Setting Up and Maintaining Message Errors....216
Pages Used to Set Up and Maintain Message Errors....217
Creating Error-Correction Pages....217
Setting Up the Error Handling Utility....218
Setting Up Row Security....220
Setting Up Workflow Notification in PeopleSoft Application Designer....220
Testing the Error Handling Utility....221
Correcting Message Errors....221
Understanding the Workflow Notification Process....222
Correcting Message Errors....222

**Chapter 20**
**Using the Effective Date Publish Utility....223**
Understanding the Effective Date Publish Utility....223
Performing a Full Data Publish of Current Effective Data....224
Pages Used for Full Data Publish of Current Effective Data....225
Creating Effective-Dated Messages....225
Creating the Service Operation....226
Defining the Node and Target Connector....226
Defining Chunking Rules and Ordering Views....227
Creating Publish Rule Definitions....227
Creating Run Controls for the Full Data Publish Program....228
Defining Routing....229
Publishing Incremental Messages of Current Effective Data....229


Copyright © 1992-2006, Oracle. All rights reserved.

Creating Service Operations for Publishing Incremental Messages of Current Effective Data . . . . . 230
Creating Subscription Processes That Open the Generic Effective-Dated Delay Function . . . . . 230
Publishing Effective-Dated Rows from the Delay Table . . . . . 230
Understanding Effective-Dated Row Publishing . . . . . 230
Page Used to Run the Effective Date Publish Utility . . . . . 231
Running the Effective Date Publish Utility . . . . . 231

**Chapter 21**
**Using the Flat File Utility . . . . . 233**
Understanding the Flat File Utility . . . . . 233
Processing Inbound Flat Files . . . . . 234
Initiating File Processing . . . . . 235
Pages Used to Initiate File Processing . . . . . 235
Setting Up Inbound Flat File Processing . . . . . 235
Initiating Inbound Flat File Processing . . . . . 237
Testing Inbound Flat File Processing . . . . . 238

**Chapter 22**
**Using the XML Schema Utility . . . . . 239**
Understanding the XML Schema Utility . . . . . 239
Generating the XML Schema . . . . . 239
Page Used to Generate XML Schema . . . . . 239
Generating the XML Schema . . . . . 239
Interpreting Sample Output . . . . . 240

**Appendix A**
**Integration Point Naming Standards . . . . . 249**
Standard Action and Event Verbs . . . . . 249
Standard Business Object Nouns . . . . . 252

**Appendix B**
**PeopleSoft Design Patterns . . . . . 255**
List of Design Patterns . . . . . 255

Copyright © 1992-2006, Oracle. All rights reserved.

**Glossary of PeopleSoft Enterprise Terms……259**

**Index ……285**


Copyright © 1992-2006, Oracle. All rights reserved.

# About This PeopleBook Preface

PeopleSoft Enterprise PeopleBooks provide you with the information that you need to implement and use PeopleSoft Enterprise applications from Oracle.

This preface discusses:

- PeopleSoft Enterprise application prerequisites.
- Application fundamentals.
- Documentation updates and printed documentation.
- Additional resources.
- Typographical conventions and visual cues.
- Comments and suggestions.
- Common elements in PeopleBooks.

---

**Note.** PeopleBooks document only elements, such as fields and check boxes, that require additional explanation. If an element is not documented with the process or task in which it is used, then either it requires no additional explanation or it is documented with common elements for the section, chapter, PeopleBook, or product line. Elements that are common to all PeopleSoft Enterprise applications are defined in this preface.

---

## PeopleSoft Enterprise Application Prerequisites

To benefit fully from the information that is covered in these books, you should have a basic understanding of how to use PeopleSoft Enterprise applications.

You might also want to complete at least one introductory training course, if applicable.

You should be familiar with navigating the system and adding, updating, and deleting information by using PeopleSoft Enterprise menus, pages, or windows. You should also be comfortable using the World Wide Web and the Microsoft Windows or Windows NT graphical user interface.

These books do not review navigation and other basics. They present the information that you need to use the system and implement your PeopleSoft Enterprise applications most effectively.

## Application Fundamentals

Each application PeopleBook provides implementation and processing information for your PeopleSoft Enterprise applications.

For some applications, additional, essential information describing the setup and design of your system appears in a companion volume of documentation called the application fundamentals PeopleBook. Most product lines have a version of the application fundamentals PeopleBook. The preface of each PeopleBook identifies the application fundamentals PeopleBooks that are associated with that PeopleBook.

Copyright © 1992-2006, Oracle. All rights reserved.

The application fundamentals PeopleBook consists of important topics that apply to many or all PeopleSoft Enterprise applications. Whether you are implementing a single application, some combination of applications within the product line, or the entire product line, you should be familiar with the contents of the appropriate application fundamentals PeopleBooks. They provide the starting points for fundamental implementation tasks.

# Documentation Updates and Printed Documentation

This section discusses how to:

- Obtain documentation updates.
- Download and order printed documentation.

## Obtaining Documentation Updates

You can find updates and additional documentation for this release, as well as previous releases, on Oracle's PeopleSoft Customer Connection website. Through the Documentation section of Oracle's PeopleSoft Customer Connection, you can download files to add to your PeopleBooks Library. You'll find a variety of useful and timely materials, including updates to the full line of PeopleSoft Enterprise documentation that is delivered on your PeopleBooks CD-ROM.

**Important!** Before you upgrade, you must check Oracle's PeopleSoft Customer Connection for updates to the upgrade instructions. Oracle continually posts updates as the upgrade process is refined.

### See Also

Oracle's PeopleSoft Customer Connection, http://www.oracle.com/support/support_peoplesoft.html

## Downloading and Ordering Printed Documentation

In addition to the complete line of documentation that is delivered on your PeopleBook CD-ROM, Oracle makes PeopleSoft Enterprise documentation available to you via Oracle's website. You can:

- Download PDF files.
- Order printed, bound volumes.

### Downloading PDF Files

You can download PDF versions of PeopleSoft Enterprise documentation online via the Oracle Technology Network. Oracle makes these PDF files available online for each major release shortly after the software is shipped.

See Oracle Technology Network, http://www.oracle.com/technology/documentation/psftent.html.

### Ordering Printed, Bound Volumes

You can order printed, bound volumes of selected documentation via the Oracle Store.

See Oracle Store, http://oraclestore.oracle.com/OA_HTML/ibeCCtpSctDspRte.jsp?section=14021


Copyright © 1992-2006, Oracle. All rights reserved.

# Additional Resources

The following resources are located on Oracle's PeopleSoft Customer Connection website:

| Resource | Navigation |
|---|---|
| Application maintenance information | Updates + Fixes |
| Business process diagrams | Support, Documentation, Business Process Maps |
| Interactive Services Repository | Support, Documentation, Interactive Services Repository |
| Hardware and software requirements | Implement, Optimize + Upgrade; Implementation Guide; Implementation Documentation and Software; Hardware and Software Requirements |
| Installation guides | Implement, Optimize + Upgrade; Implementation Guide; Implementation Documentation and Software; Installation Guides and Notes |
| Integration information | Implement, Optimize + Upgrade; Implementation Guide; Implementation Documentation and Software; Pre-Built Integrations for PeopleSoft Enterprise and JD Edwards EnterpriseOne Applications |
| Minimum technical requirements (MTRs) | Implement, Optimize + Upgrade; Implementation Guide; Supported Platforms |
| Documentation updates | Support, Documentation, Documentation Updates |
| PeopleBooks support policy | Support, Support Policy |
| Prerelease notes | Support, Documentation, Documentation Updates, Category, Release Notes |
| Product release roadmap | Support, Roadmaps + Schedules |
| Release notes | Support, Documentation, Documentation Updates, Category, Release Notes |
| Release value proposition | Support, Documentation, Documentation Updates, Category, Release Value Proposition |
| Statement of direction | Support, Documentation, Documentation Updates, Category, Statement of Direction |
| Troubleshooting information | Support, Troubleshooting |
| Upgrade documentation | Support, Documentation, Upgrade Documentation and Scripts |

Copyright © 1992-2006, Oracle. All rights reserved.

# Typographical Conventions and Visual Cues

This section discusses:

- Typographical conventions.
- Visual cues.
- Country, region, and industry identifiers.
- Currency codes.

## Typographical Conventions

This table contains the typographical conventions that are used in PeopleBooks:

| Typographical Convention or Visual Cue | Description |
|---|---|
| **Bold** | Indicates PeopleCode function names, business function names, event names, system function names, method names, language constructs, and PeopleCode reserved words that must be included literally in the function call. |
| *Italics* | Indicates field values, emphasis, and PeopleSoft Enterprise or other book-length publication titles. In PeopleCode syntax, italic items are placeholders for arguments that your program must supply.<br>We also use italics when we refer to words as words or letters as letters, as in the following: Enter the letter *O*. |
| KEY+KEY | Indicates a key combination action. For example, a plus sign (+) between keys means that you must hold down the first key while you press the second key. For ALT+W, hold down the ALT key while you press the W key. |
| `Monospace font` | Indicates a PeopleCode program or other code example. |
| “ ” (quotation marks) | Indicate chapter titles in cross-references and words that are used differently from their intended meanings. |
| . . . (ellipses) | Indicate that the preceding item or series can be repeated any number of times in PeopleCode syntax. |
| { } (curly braces) | Indicate a choice between two options in PeopleCode syntax. Options are separated by a pipe ( \| ). |


Copyright © 1992-2006, Oracle. All rights reserved.

| Typographical Convention or Visual Cue | Description |
|---|---|
| [ ] (square brackets) | Indicate optional items in PeopleCode syntax. |
| & (ampersand) | When placed before a parameter in PeopleCode syntax, an ampersand indicates that the parameter is an already instantiated object.<br>Ampersands also precede all PeopleCode variables. |

# Visual Cues

PeopleBooks contain the following visual cues.

## Notes

Notes indicate information that you should pay particular attention to as you work with the PeopleSoft Enterprise system.

**Note.** Example of a note.

If the note is preceded by *Important!,* the note is crucial and includes information that concerns what you must do for the system to function properly.

**Important!** Example of an important note.

## Warnings

Warnings indicate crucial configuration considerations. Pay close attention to warning messages.

***Warning!*** Example of a warning.

## Cross-References

PeopleBooks provide cross-references either under the heading "See Also" or on a separate line preceded by the word *See.* Cross-references lead to other documentation that is pertinent to the immediately preceding documentation.

# Country, Region, and Industry Identifiers

Information that applies only to a specific country, region, or industry is preceded by a standard identifier in parentheses. This identifier typically appears at the beginning of a section heading, but it may also appear at the beginning of a note or other text.

Example of a country-specific heading: "(FRA) Hiring an Employee"

Example of a region-specific heading: "(Latin America) Setting Up Depreciation"

## Country Identifiers

Countries are identified with the International Organization for Standardization (ISO) country code.

Copyright © 1992-2006, Oracle. All rights reserved.

### Region Identifiers

Regions are identified by the region name. The following region identifiers may appear in PeopleBooks:

- Asia Pacific
- Europe
- Latin America
- North America

### Industry Identifiers

Industries are identified by the industry name or by an abbreviation for that industry. The following industry identifiers may appear in PeopleBooks:

- USF (U.S. Federal)
- E&G (Education and Government)

## Currency Codes

Monetary amounts are identified by the ISO currency code.

# Comments and Suggestions

Your comments are important to us. We encourage you to tell us what you like, or what you would like to see changed about PeopleBooks and other Oracle reference and training materials. Please send your suggestions to your product line documentation manager at Oracle Corporation, 500 Oracle Parkway, Redwood Shores, CA 94065, U.S.A. Or email us at appsdoc@us.oracle.com.

While we cannot guarantee to answer every email message, we will pay careful attention to your comments and suggestions.

# Common Elements Used in PeopleBooks

| | |
|---|---|
| **As of Date** | The last date for which a report or process includes data. |
| **Business Unit** | An ID that represents a high-level organization of business information. You can use a business unit to define regional or departmental units within a larger organization. |
| **Description** | Enter up to 30 characters of text. |
| **Effective Date** | The date on which a table row becomes effective; the date that an action begins. For example, to close out a ledger on June 30, the effective date for the ledger closing would be July 1. This date also determines when you can view and change the information. Pages or panels and batch processes that use the information use the current row. |


Copyright © 1992-2006, Oracle. All rights reserved.

| | |
|---|---|
| **Once, Always,** and **Don't Run** | Select Once to run the request the next time the batch process runs. After the batch process runs, the process frequency is automatically set to Don't Run.<br>Select Always to run the request every time the batch process runs.<br>Select Don't Run to ignore the request when the batch process runs. |
| **Process Monitor** | Click to access the Process List page, where you can view the status of submitted process requests. |
| **Report Manager** | Click to access the Report List page, where you can view report content, check the status of a report, and see content detail messages (which show you a description of the report and the distribution list). |
| **Request ID** | An ID that represents a set of selection criteria for a report or process. |
| **Run** | Click to access the Process Scheduler request page, where you can specify the location where a process or job runs and the process output format. |
| **SetID** | An ID that represents a set of control table information, or TableSets. TableSets enable you to share control table information and processing options among business units. The goal is to minimize redundant data and system maintenance tasks. When you assign a setID to a record group in a business unit, you indicate that all of the tables in the record group are shared between that business unit and any other business unit that also assigns that setID to that record group. For example, you can define a group of common job codes that are shared between several business units. Each business unit that shares the job codes is assigned the same setID for that record group. |
| **Short Description** | Enter up to 15 characters of text. |
| **User ID** | An ID that represents the person who generates a transaction. |

Copyright © 1992-2006, Oracle. All rights reserved.

Copyright © 1992-2006, Oracle. All rights reserved.

# PeopleSoft Enterprise Components Preface

This preface provides an overview of PeopleSoft Enterprise Component documentation included in this PeopleBook.

## Overview of PeopleSoft Enterprise Components

PeopleSoft Enterprise Components include a variety of features that are generally available as common objects or components to all product lines delivered by PeopleSoft. This PeopleBook include the following parts:

- *Getting Started With Enterprise Components.* Includes getting started sections describing each of the product areas.
- *PeopleSoft Directory Interface.* Describes how to enhance the integration between PeopleSoft and your LDAP directory service by providing a single, centralized user profile for PeopleSoft and non-PeopleSoft applications. The flexible directory structure and schema leverage the LDAP security model to control access to PeopleSoft applications.
- *Common Objects and Components.* Describes features generally available as common objects or components to all product lines, including interactive reports, currency and market rate components, datasets, and the credit card interface.
- *Enterprise Integration.* Describes features related to integration from one PeopleSoft product to another, and integration between PeopleSoft and third-party products.

**Note.** Some of the page elements and colors that your product uses may differ from the screen shots presented in this PeopleBook. This book uses a generic style sheet for the purposes of illustration only.

Copyright © 1992-2006, Oracle. All rights reserved.

Copyright © 1992-2006, Oracle. All rights reserved.

# PART 1

# Getting Started With PeopleSoft Enterprise Components

**Chapter 1**
**Getting Started With PeopleSoft Enterprise Components**

# CHAPTER 1

# Getting Started With PeopleSoft Enterprise Components

This chapter provides information about getting started in the following PeopleSoft Enterprise Component areas:

- PeopleSoft Directory Interface
- Common Objects and Components
- Enterprise Integration

## Getting Started With PeopleSoft Directory Interface

Setting up to use the PeopleSoft Directory Interface requires you access components in PeopleSoft PeopleTools to define and activate nodes and transactions, create authentication and user profile maps, and so on. You also access components in PeopleSoft Enterprise Components to configure and load the directory.

See Chapter 2, "Using PeopleSoft Directory Interface," page 7.

### Implementation Guidelines

Consider these guidelines for best results.

- *LDAP Searches*. Some LDAP searches may generate LDAP referrals to other servers participating in your directory. You must be able to ping by hostname all servers in the directory from the application server. If any server is unreachable by hostname from the application server, you can add a line for the server to the hosts. Your directory information tree must have user entries at the leaf level. This is required when an entry needs to be moved from one branch to another. The entry needs to be at the leaf level so that the system can read user attributes, one of which is the password. file on the application server.
- *Entry Limit*. In the directory, configure the entry limit value to be larger than the number of rows that you expect will be returned. The default value is usually not sufficient.
- *Directory Tree.* Your directory information tree must have user entries at the leaf level. This is required when an entry needs to be moved from one branch to another. The entry needs to be at the leaf level so that the system can read user attributes, one of which is the password.
- The following items apply to implementations that use Microsoft Active Directory:
  - The registry key HKLM\System\CurrentControlSet\Services\NTDS\Parameters\Schema Update Allowed must be present and set to a nonzero DWORD on the Active Directory FSMO Role Owner.
  - When creating structural object classes in Microsoft Active Directory, you need to specify containment. PsftJobs can be children of the following classes of objects only: builtinDomain, organizationalUnit, and domainDNS.

Copyright © 1992-2006, Oracle. All rights reserved.

- You must add the server names in the Directory Setup component as they appear on the DNSHost Name attribute on the server entries under the CN=Sites entry.

# Getting Started With Common Objects and Components

PeopleSoft Enterprise Components provides objects and components that can be used by various PeopleSoft applications. For set up or implementation information of the following components, refer to each chapter.

See Chapter 5, “Using Datasets,” page 63.

See Chapter 6, “Using Interactive Reports,” page 69.

See Chapter 7, “Setting Up the Credit Card Interface,” page 77.

# Getting Started With Enterprise Integration

PeopleSoft enterprise integration technology includes various features and utilities. The elements you use will depend on your PeopleSoft application.

If you are using integration points, the following sections may be helpful:

See Appendix A, “Integration Point Naming Standards,” page 249.

See Appendix B, “PeopleSoft Design Patterns,” page 255.

See Chapter 17, “Activating Integration Points,” Setting Up PeopleSoft Integration Points, page 183.

See Chapter 17, “Activating Integration Points,” Setting Up Related Languages, page 186.

See Chapter 19, “Using the Error Handling Utility,” Setting Up and Maintaining Message Errors, page 216.

 Copyright © 1992-2006, Oracle. All rights reserved.

# PART 2

# PeopleSoft Directory Interface

**Chapter 2**
**Using PeopleSoft Directory Interface**

**Chapter 3**
**Reviewing Directory Data and Generating Reports**

# CHAPTER 2

# Using PeopleSoft Directory Interface

This chapter provides an overview of PeopleSoft Directory Interface and discusses how to:

- Define and configure the directory.
- Set up directory authentication.
- Load the directory schema.
- Set up mappings.
- Set up entry membership rules.
- Load data into the directory.

## Understanding PeopleSoft Directory Interface

PeopleSoft Directory Interface utilizes Lightweight Directory Access Protocol (LDAP) directory services to authenticate users of PeopleSoft applications.

PeopleSoft Directory Interface provides additional mappings and integration points, such as messages, that enable PeopleSoft data and LDAP data to stay synchronized. Most directory data, such as user ID, name, and email address, is also maintained in your PeopleSoft database. When you use PeopleSoft Directory Interface, you make selected PeopleSoft data available to the directory, and you maintain the data in PeopleSoft.

When information changes in the PeopleSoft database, PeopleSoft Directory Interface captures that updated information and automatically updates the equivalent information in the directory server, or it writes the updates to a file for you to apply at another time.

### Understanding Data Mapping

PeopleSoft information is stored in tables according to a relational model. The information in your LDAP directory is stored in trees according to a hierarchical model. You use PeopleSoft Directory Interface to map selected PeopleSoft data to corresponding data in the directory service. When PeopleSoft Directory Interface receives user data from the PeopleSoft database, it can map the data objects to the corresponding objects in the directory.

For PeopleSoft Directory Interface to map PeopleSoft information to your directory, it needs information about the directory hierarchical structure, or *directory information tree.*

Entries are made up of a *distinguished name* (DN) and attribute/value pairs. The distinguished name identifies an entry's position in the tree, and the attributes hold the data that make up the entry.

Available attributes for an object class entry are specified in the directory schema. You must load the schema into the Directory Interface before you can map PeopleSoft data to the directory.

Copyright © 1992-2006, Oracle. All rights reserved.

PeopleSoft Directory Interface mapping tables map LDAP attributes to PeopleSoft messages. Each message contains selected information about a PeopleSoft record and its fields.

**Note.** Refer to PeopleSoft application documentation for information about specific messages delivered by PeopleSoft applications.

### Understanding Data Synchronization

Once you have loaded PeopleSoft data into your LDAP directory, you can synchronize the data. To do this, use one of the following options:

- PeopleSoft Business Interlinks

  PeopleSoft Business Interlinks updates the data in real time, so that your directory information is always synchronized with PeopleSoft.

- LDAP Data Interchange Format (LDIF) files.

  You can load LDIF files as needed or defined by your system.

**Note.** The application server needs to be configured for receiving messages.

### Delivered Business Interlinks

PeopleSoft delivers the following business interlinks with PeopleSoft Directory Interface:

| | |
|---|---|
| **EO_DS_ADD** | Adds a new entry to the directory by creating a distinguished name and its corresponding attributes. |
| **EO_DS_BIND** | Authenticates the information exchanged between the database and the directory. |
| **EO_DS_DEL** | Deletes an entry from the directory. |
| **EO_DS_MODDN** | Renames a directory entry. Changes its distinguished name by renaming the actual entry or changing its position in the directory entry. |
| **EO_DS_MODIFY** | Changes the attributes of an entry. |
| **EO_DS_SEARCH** | Searches for directory entries and their corresponding attributes. |

See *Enterprise PeopleTools 8.48 PeopleBook: PeopleSoft Business Interlinks*

# Overview of Using PeopleSoft Directory Interface

The section briefly describes the steps needed to use PeopleSoft Directory Interface, including:

- Setting up in PeopleSoft Application Designer and PeopleSoft Integration Broker.
- Using the Directory Configurations component

 Copyright © 1992-2006, Oracle. All rights reserved.

# Setting Up in PeopleSoft Application Designer and PeopleSoft Integration Broker

Perform the following steps in PeopleSoft Application Designer and PeopleSoft Integration Broker.

## Setting Up in PeopleSoft Application Designer

Access PeopleSoft Application Designer.

- Create authentication and user profile maps as needed.

  If you are going to authenticate users with the directory server, a PeopleSoft user profile is required—that is, a row in the PSOPRDEFN table where PeopleSoft user information is stored. In this context, you "cache" LDAP user information inside your PeopleSoft system. Properties you specify in the Mandatory and Optional Properties pages of the Mappings component are the columns in PSOPRDEFN that the system populates with values from your directory server. PeopleSoft applications use this cache of user information, not your directory server. Whenever a transaction requires user information, the application refers to the local PSOPRDEFN table instead of querying the directory server.

- Add Signon PeopleCode.

  Directory authentication requires Signon PeopleCode be enabled and configured with proper permissions. After a user signs onto the system and the Signon PeopleCode executes, PeopleSoft creates a row for the user in the user definition table by retrieving the LDAP information and creating a local cache. Signon PeopleCode maintains this row automatically and any changes made in the directory server are reproduced in the local cache. Using the Mappings component, set up mappings. To keep the data synchronized, you must map PeopleSoft data to the equivalent directory objects. PeopleSoft Directory Interface then associates the fields in the message to the attributes in the directory and updates the selected directory attributes with the field data from the message.

- Activate the DSCHNL channel.

  Open the message channel and select Run.

See *Enterprise PeopleTools 8.48 PeopleBook: PeopleSoft Application Designer*

## Setting Up in PeopleSoft Integration Broker

Access PeopleSoft Integration Broker.

- Activate a relevant node.

  This node should be the default local node.

- Define a service operation.

  Note that the service operations, and messages to include in the service operations, depend on the application. For example, in an HR implementation, you might want to include messages such as Dept, Location, Person, and Job in a service operation, in addition to core messages such as DSMINPUT.

See *Enterprise PeopleTools 8.48 PeopleBook: PeopleSoft Integration Broker*

# Using the Directory Configurations Component

Access Directory Configurations component (PSDSSETUP) from the browser menu.

- Using the Directory Configurations component, configure the directory.

  Enter appropriate connection information such as the server name (DNS or IP address) and the listening port number, the user DN, and associated password.

Copyright © 1992-2006, Oracle. All rights reserved.

- Using the Schema Management page, select names of Object Classes and Attribute Types and then cache the schema.
- To keep the data synchronized, you must map PeopleSoft data to the equivalent directory objects.

  Set up mappings using the Mappings component. Once this is complete, PeopleSoft Directory Interface associates the fields in the message to the attributes in the directory and updates the selected directory attributes with the field data from the message.
- Using the Membership Rules component, create rules and memberships, if desired.
- Load data in the directory.
- Set directory search criteria.

  Enter search parameters to query the directory and view the results.

# Common Elements Used in This Chapter

| | |
|---|---|
| **Directory ID** | Unique identifier for the directory. |
| **Description** | A brief description of the directory. |
| **Directory Product** | Select the directory product from the drop-down list box. |
| **Default Connect DN** | Displays the connect distinguished name associated with the directory ID that you selected. Use this ID to connect to the directory server. |
| **Password** | Password to access the directory. |
| **LDAP Server** | The server name where the directory resides. |
| **Port** | The LDAP server port associated with the LDAP server you select. |
| **SSL Port** | The secure socket layer port. |

# Defining and Configuring the Directory

Use the Directory Configurations component (PSDSSETUP) to define and configure the directory connection. In this section, we discuss how to:

- Configure the directory connection.
- Cache the schema.
- Delete the directory configuration.

### See Also

*Enterprise PeopleTools 8.48 PeopleBook: Security Administration,* "Employing LDAP Directory Services," Configuring the LDAP Directory

 Copyright © 1992-2006, Oracle. All rights reserved.

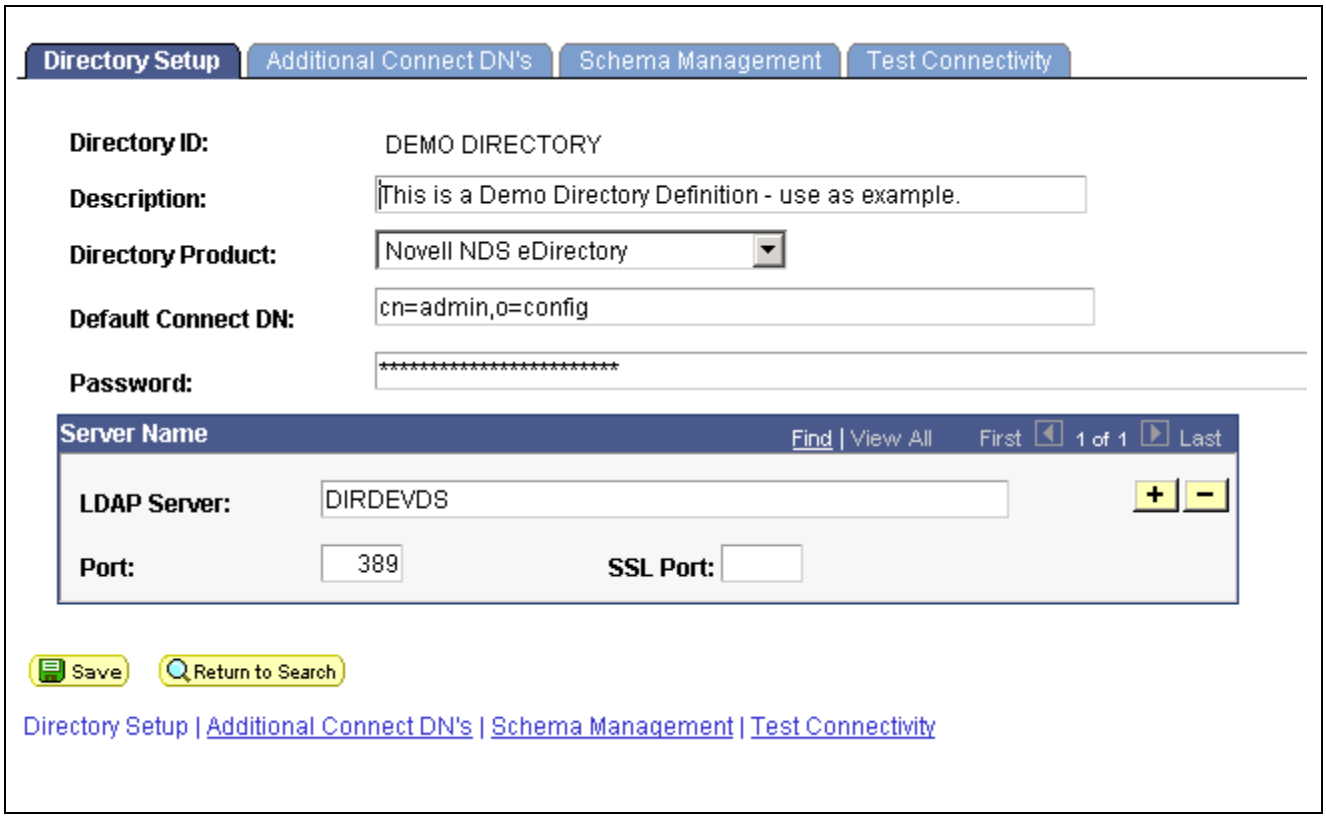
## Pages Used to Define and Configure the Directory

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Directory Setup | DSDIRSETUP | Enterprise Components, Directory Interface, Definitions, Directory Configurations, Directory Setup | Enter values to configure the directory. |
| Additional Connect DN's | DSSERVERID | Enterprise Components, Directory Interface, Definitions, Directory Configurations, Additional Connect DN's | Add values for additional connect DNs. |
| Schema Management | DSEXTINSTALL | Enterprise Components, Directory Interface, Definitions, Directory Configurations, Schema Management | Manage schema, apply schema PeopleSoft schema extensions. |
| Test Connectivity | DSSRCHRSLT | Enterprise Components, Directory Interface, Definitions, Directory Configurations, Test Connectivity | Test the directory connectivity. |
| Cache Schema | DSSCHEMACACHE | Enterprise Components, Directory Interface, Definitions, Schema Caching | Cache the schema. |
| Delete Directory | DSPURGEDIRID | Enterprise Components, Directory Interface, Definitions, Directory Deletions | Delete the directory configuration. |

## Configuring the Directory Connection

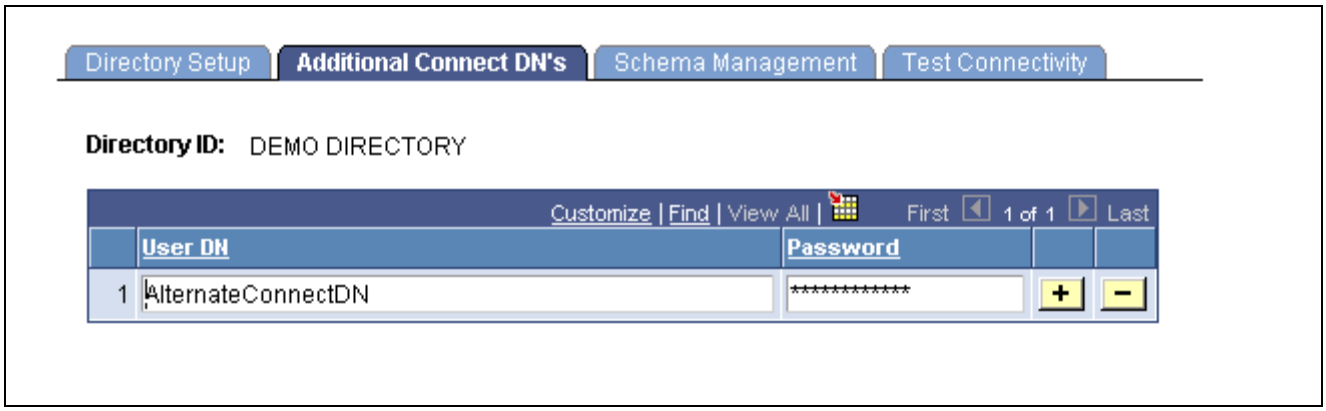
Access the Directory Setup page.

Copyright © 1992-2006, Oracle. All rights reserved.




Directory Setup

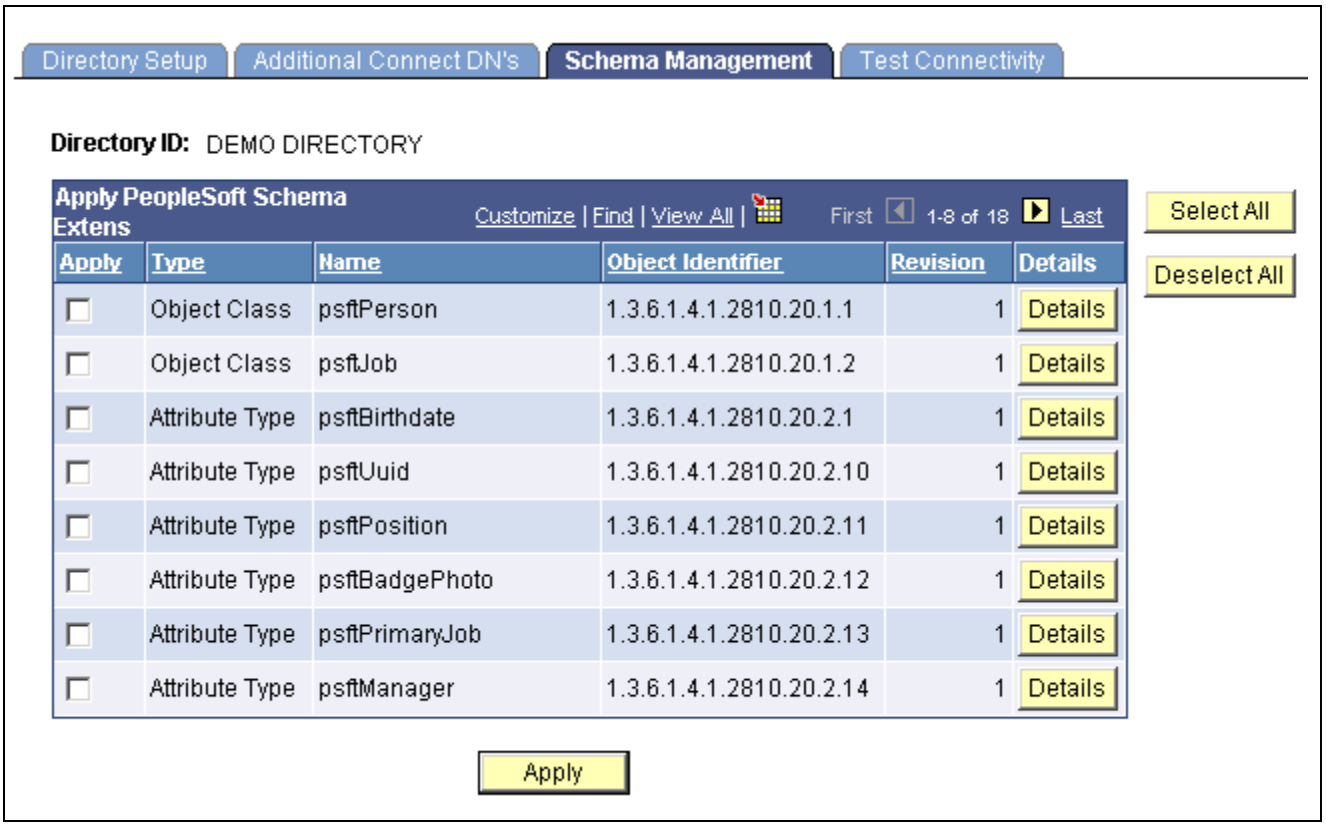
Access the Additional Connect DN's page and add more Connect DNs and passwords, if needed.



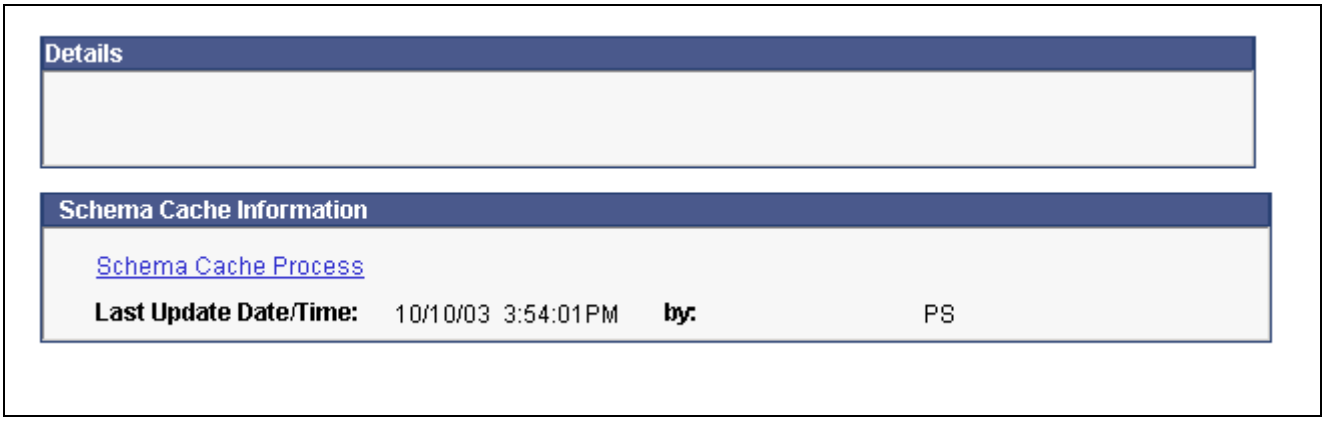

Additional Connect DN's

Access the Schema Management page.


Copyright © 1992-2006, Oracle. All rights reserved.




Schema Management page (1 of 2)



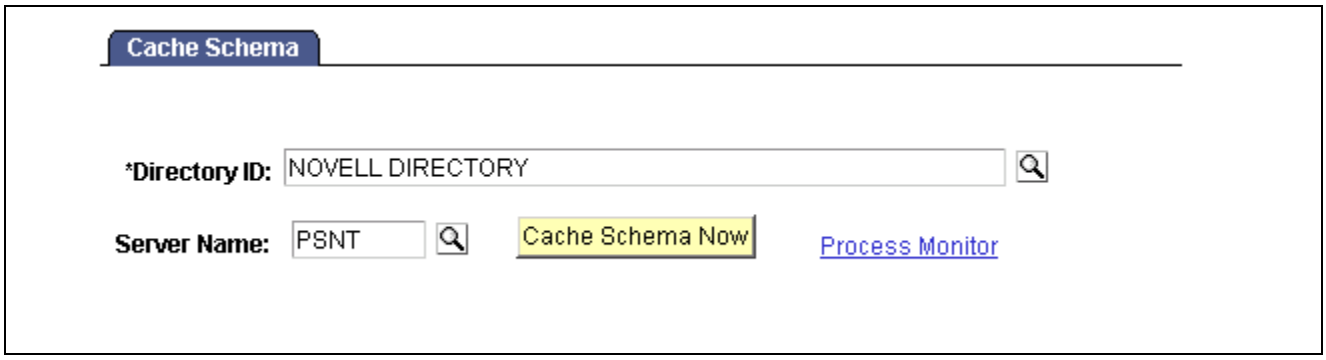

Schema Management page (2 of 2)

Activate the check boxes of those object classes or attribute types that you want applied to the cache schema.

## Caching the Schema

Access the Schema Caching page.

Copyright © 1992-2006, Oracle. All rights reserved. 

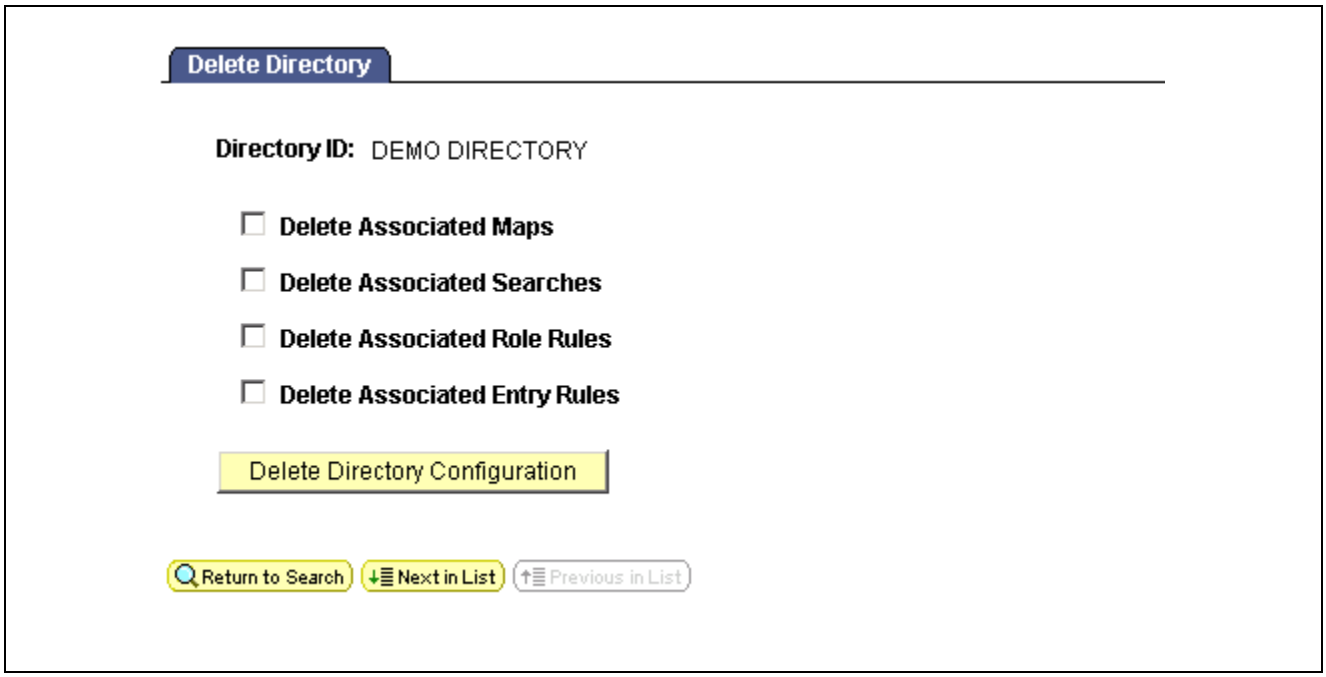



Cache Schema page

Enter the Directory ID and Server Name of the schema to be cached and click the Cache Schema Now button.

## Deleting the Directory Configuration

Access the Delete Directory page.




Delete Directory page

Select the check boxes for the desired directory configuration deletions.

# (Optional) Setting Up Directory Authentication

In this section, we discuss how to:

- Use map authentication.
- Viewing user properties.

For information on setting up authentication servers, user profile maps, and role membership rules, refer to the following documentation.


Copyright © 1992-2006, Oracle. All rights reserved.

### See Also

*Enterprise PeopleTools 8.48 PeopleBook: Security Administration,* "Employing LDAP Directory Services," Creating the Authentication Map

*Enterprise PeopleTools 8.48 PeopleBook: Security Administration,* "Employing LDAP Directory Services," Creating User Profile Maps

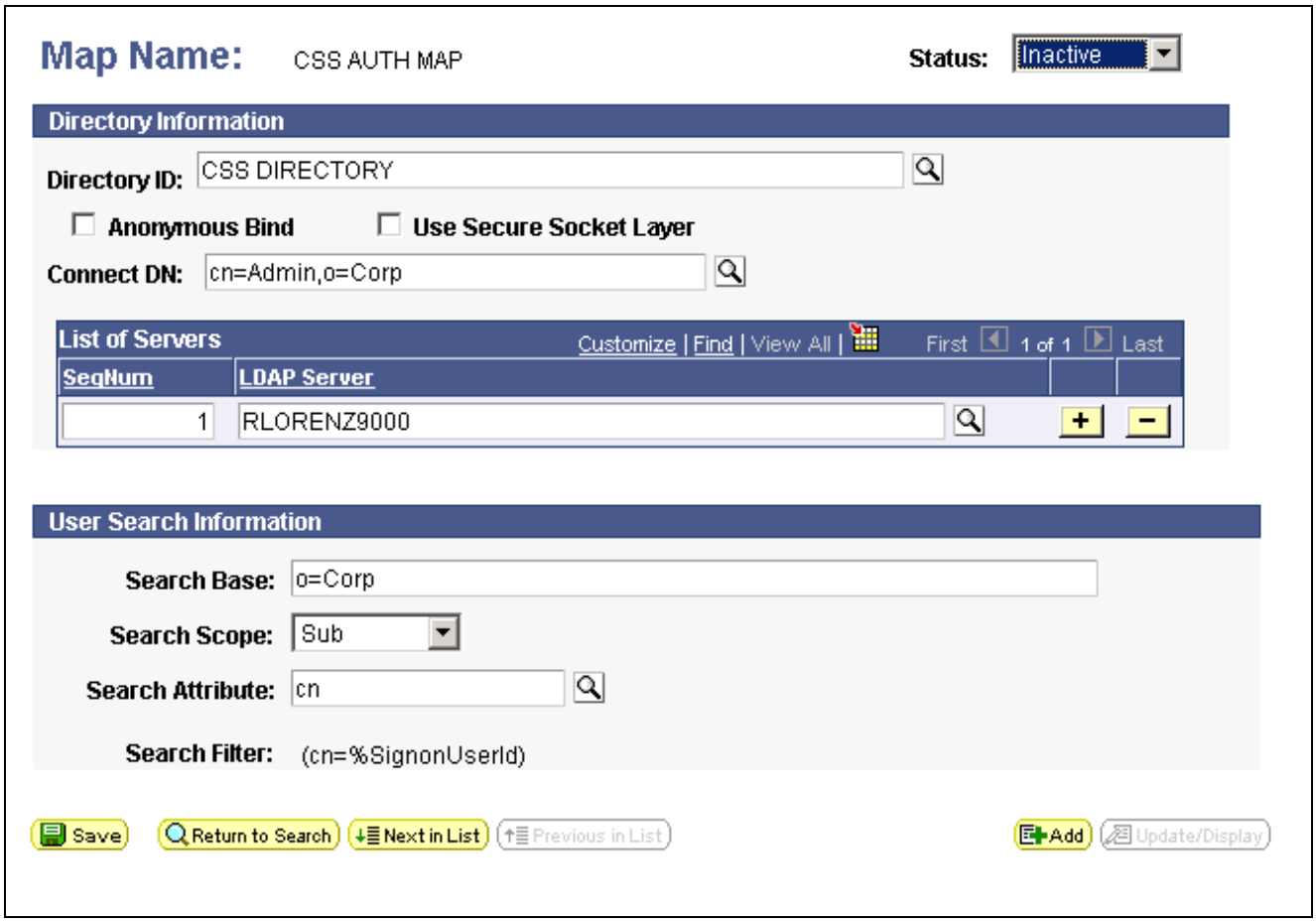
## Pages Used to Set Up Directory Authentication

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Authentication | DSSECMAPMAIN | Enterprise Components, Directory Interface, Mappings, Authentication | Create a mapping for the directory that the system relies on for authenticating users. |
| Mandatory User Properties | DSUSRPRFLMANMAP | Enterprise Components, Directory Interface, Mappings, User Profiles, Mandatory User Properties | Specify the attributes required for sign-in. You can have the system retrieve these mandatory values from the directory server, or you can enter default values. |
| Optional User Properties | DSUSRPRFLOPTMAP | Enterprise Components, Directory Interface, Mappings, User Profiles, Optional User Properties | Specify optional user properties to store in and retrieve from the directory. You can specify general, permission list, and workflow attributes. All these attributes appear in the User Profile component. |

## Using Map Authentication

Access the Authentication page.

Copyright © 1992-2006, Oracle. All rights reserved. 




Authentication page

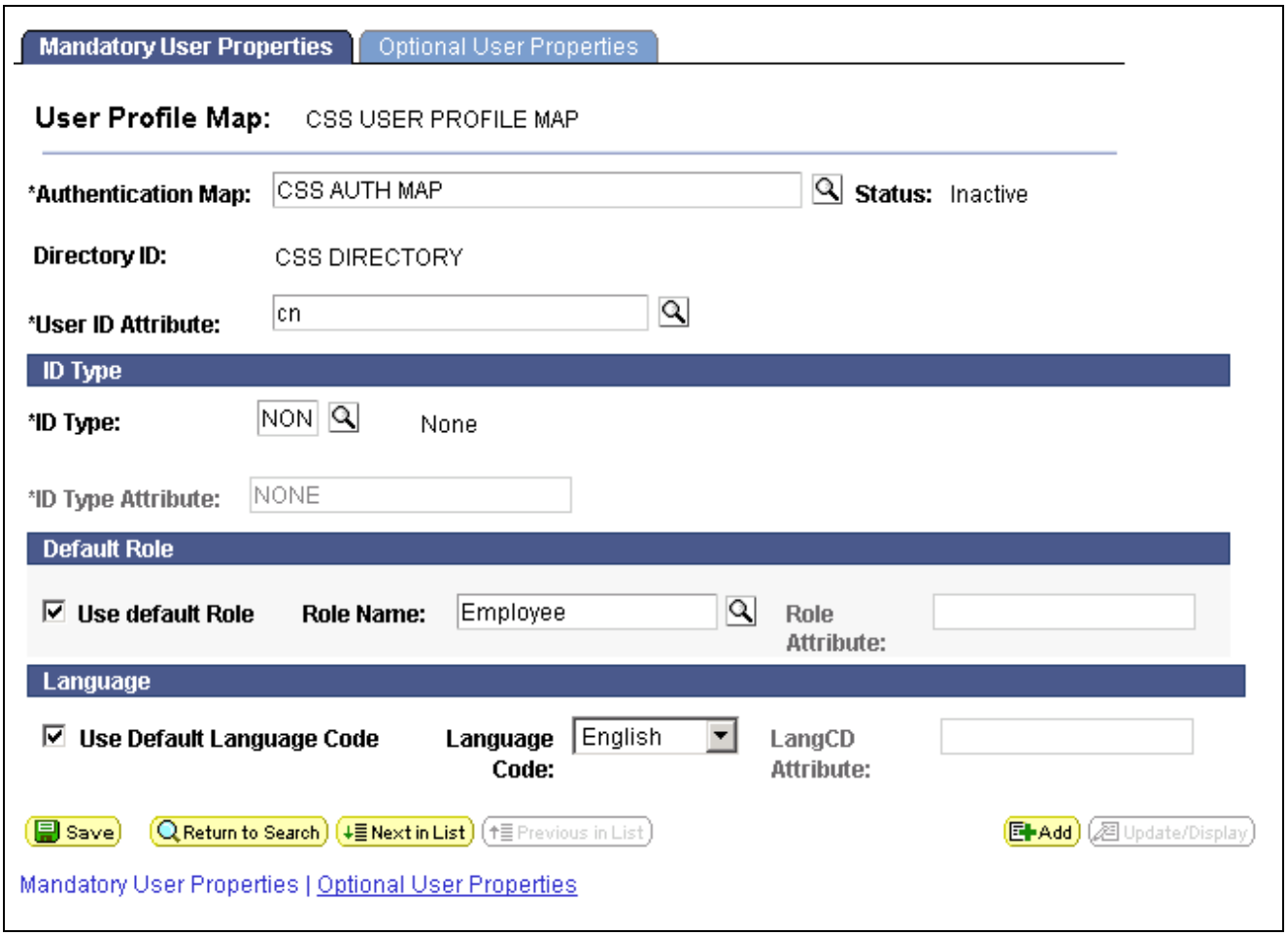
**Anonymous Bind** If directory data required for authentication and user profile maintenance is visible to an anonymous connection you may select this check box.

**Use Secure Socket Layer** Select if you are using SSL between PeopleSoft and the directory server.

## Viewing User Properties

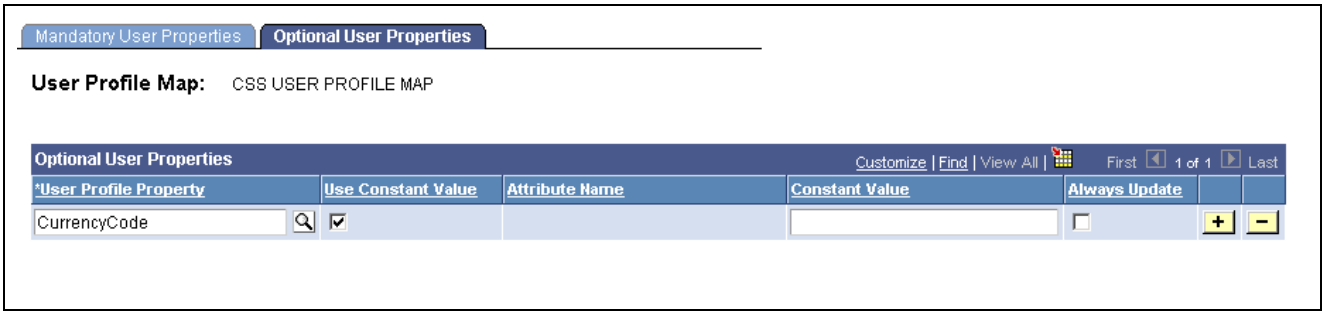
Access the Mandatory or Optional User Properties page.


Copyright © 1992-2006, Oracle. All rights reserved.




Mandatory User Properties page

Select the Authentication Map and set check boxes and field values as needed.




Optional User Properties page

# Setting Up Mappings

This section provides an overview of mapping and discusses how to:

- Create mappings.
- Modify a distinguished name.

Copyright © 1992-2006, Oracle. All rights reserved.

- Specify distinguished name details.
- Translate or perform functions with database values.
- Map PeopleSoft data to directory object class attributes.
- Locate delivered messages.

## Understanding Mappings

You map PeopleSoft data to the equivalent directory objects to keep the data synchronized. PeopleSoft Directory Interface receives PeopleSoft data from messages contained within service operation that you publish whenever there is a business event associated with the messages identified in the Directory Mapping component. Each message contains information about records and the most recent data for the record fields. Using the mapping information that you set up, PeopleSoft Directory Interface associates the fields in the message to the attributes in the directory and then updates the selected directory attributes with the field data from the message. Additionally, you can define a constant value or a PeopleCode function that returns a value to populate data used in building temporary Directory Information Trees when not all data exists for an entry.

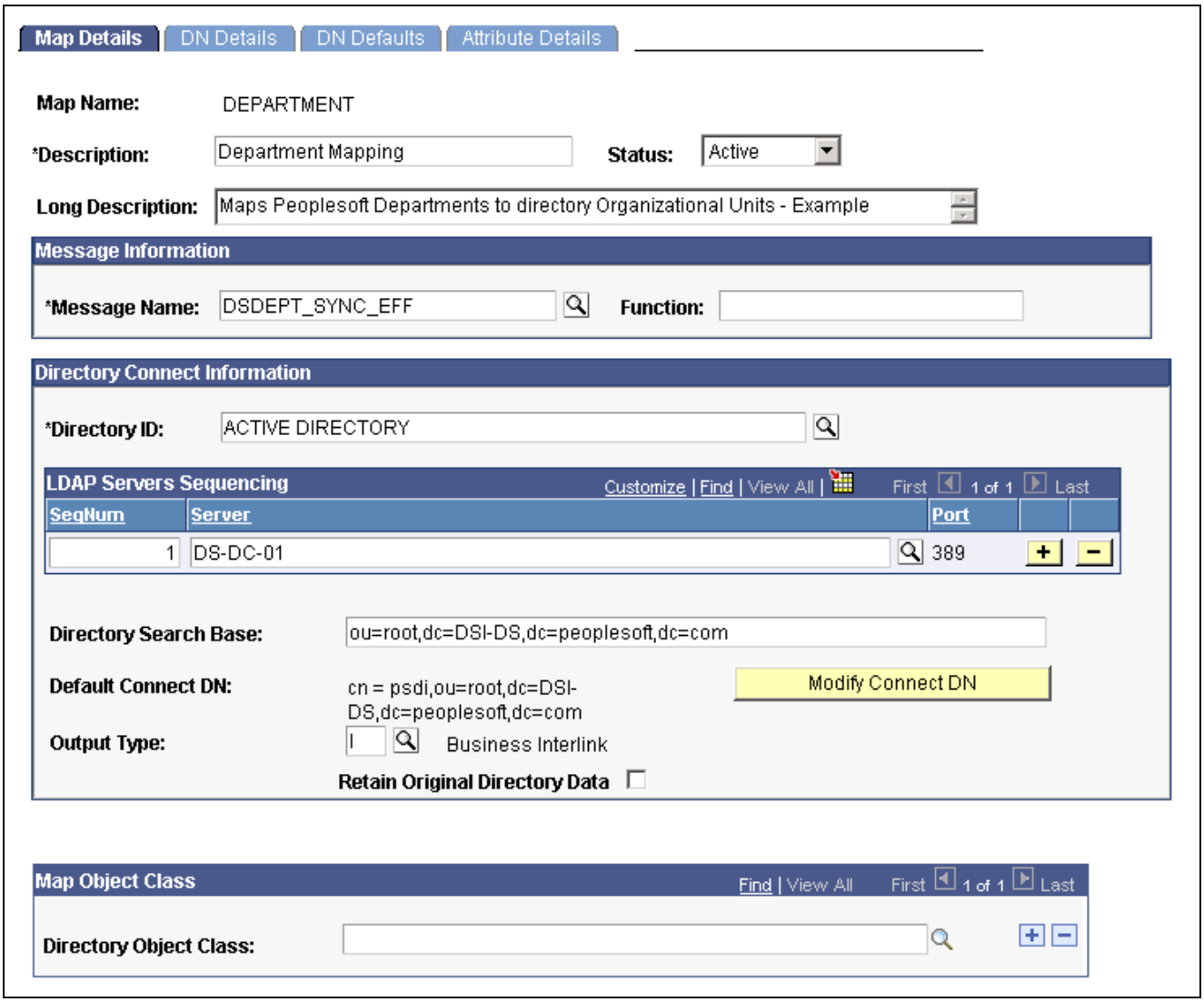
## Pages Used to Set Up Mappings

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Map Details | EO_DSMAP | Enterprise Components, Directory Interface, Mappings, Directory maps, Map Details | Set up a mapping and enter the data relationship details between PeopleSoft data and directory data. |
| Modify Connect DN - Directory Interface | EO_DSUSERDN | Click the Modify Connect DN button on the Map Details page. | Allows you to modify the Connect DN. |
| DN Details | EO_DSDN | Enterprise Components, Directory Interface, Mappings, Directory maps, DN Details | Set up the relationship between the data contained in the message that you selected on the Message Details page and the directory entry's distinguished name. |
| DN Defaults | EODS_DN_DEFAULTS | Enterprise Components, Directory Interface, Mappings, Directory maps, DN Defaults | Define a constant value or PeopleCode function that returns values that populate the blank values on the Directory Information Tree. |
| Attribute Details | EO_DSATTRIB | Enterprise Components, Directory Interface, Mappings, Directory maps, Attribute Details | Set up the relationship between the data in the message that you selected on the Message Details page and the directory object class attributes. |

## Creating Mappings

Access the Map Details page.

 Copyright © 1992-2006, Oracle. All rights reserved.




Map Details page

**Status** Select the appropriate status from the following values.

- *Active:* The map is active and ready to be used.
- *Inactive:* The map is not ready to be used.
- *Remote:* Not used at this time, and may appear unavailable.

## Message Information

**Message Name** Select the message to associate with this mapping. The message contains the PeopleSoft records and fields that have the data that you want to associate with the attributes that make up the directory entry that you select in the Directory Connect Information group box. For example, if you select the output − DEPTID object class, select the department (DSDEPT_SYNC) message because it contains the fields relevant to the department object class.

**Function** Enter the name of the PeopleCode function that you want to run using this message as an input parameter. The function can use any of the fields

Copyright © 1992-2006, Oracle. All rights reserved. 

contained in the message to produce an output value for one or more of the fields that you map. This enables you to use a field in a function without mapping to it directly. For example, if you want the employee ID value sent to the directory to be a value combining the employee ID and the salary code, enter a function that produces that value. You then only have to map to the EmplID field to insert the derived employee ID in the directory.

### Directory Connect Information

**SeqNum** (directory sequence number) Indicate the order in which the server should be used when the system processes this mapping. If the first server is unavailable, the system attempts to access the other servers in sequence until it finds an available one. If you are using multiple servers, this enables you to distribute the load across servers.

**Directory Search Base** Enter a directory search base. The search base is the entry in the directory information tree from which the system begins a search relating to this mapping. For example, if, on the Attribute Details page, you select to have a field value updated indirectly, PeopleSoft Directory Interface searches for and updates all instances of that field in entries from that point in the information tree down.

**Modify Connect DN** Click to access the Modify Connect DN - Directory Interface page.

**Output Type** Select the method that the system should use to send the mapped data to the directory data. Select *I* to output data to the directory directly through a business interlink. Select *F* to output data to an LDAP Data Interchange Format (LDIF) file to be manually updated in the directory.

Use the same output type for all your mappings to keep data consistent in the directory.

**Retain Original Directory Data** When you move data in your directory using the Directory Interface, the Directory Interface copies the data to the new location and then deletes the old version. Select this check box to preserve the original data. You can select this check box at a later date provided that you do it before the data move.

---

**Note.** Select this check box if your directory contains binary data. Move the binary data with your directory administrative tool.

---

### Map Object Class

**Directory Object Class** Select one or more directory object classes. The object classes that you select determine the attributes you can map to PeopleSoft data.

## Modifying the Distinguished Name

Access the Modify Connect DN - Directory Interface page.

**Use Default (Admin) DN** (use default [administrative] distinguished name) Select to use the default connect distinguished name value that you set up in PeopleTools.

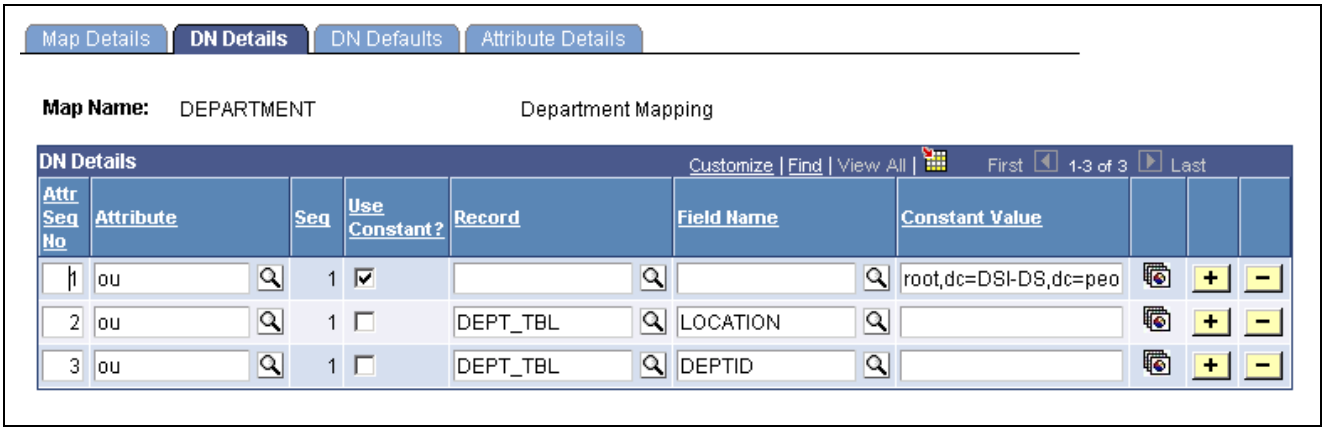
**User DN** (user distinguished name) Displays the alternative IDs that you can use to connect to the specified directory ID. You can use a user ID (and password) other then the default one listed on the Directory Setup page in PeopleTools. Because the default user ID

 Copyright © 1992-2006, Oracle. All rights reserved.

is most likely an administrative ID, this enables you to set up a more secure user ID for the scope of the mapping.

## Specifying Distinguished Name Details

Access the DN Details page.




DN Details page

Associate the data contained in the message that you selected on the Map Details page with the entry's distinguished name.

### DN Details

**Attr Seq No** (attribute sequence number) The system assigns an attribute sequence number to the attributes. Some directory attribute values consist of multiple values. The attribute sequence number distinguishes between the different attribute values and indicates to PeopleSoft Directory Interface the order in which the PeopleSoft and constant values should be assigned to the attribute.

**Attribute** Select the directory attributes associated with the mapping's distinguished name. For example, for the Department entry, map the o − Corporation first, the l − location second, and then the ou − Department attribute.

**Seq** (sequence) Enter the sequence number of the directory attribute. The directory builds the entry's distinguished name using the attributes in sequential order.

**Use Constant** and **Constant Value** Select to use the constant value that you enter in the Constant Value field to populate this attribute instead of a PeopleSoft field value.

**Record** and **Field Name** Select the name of the record that contains the PeopleSoft field and the PeopleSoft field containing the value to assign to this attribute.

Click to access the DN Attribute Function - Directory Interface page and translate database values or instruct the system to perform functions with database values.

**Note.** Use this page when constructing distinguished names across active directory multiple domains.

Copyright © 1992-2006, Oracle. All rights reserved.

### Example Entry

An entry's distinguished name is built by applying the attributes in a sequential order. The order for the department entry example would be constructed using the data in the following table:

| Sequence Number | Directory Attribute | Attribute Sequence Number | Use Constant Value | Record (Table) Name | Field Name | Constant Value |
|---|---|---|---|---|---|---|
| 1 | o | 1 | Yes | | | Corp |
| 2 | l | 1 | No | DEPT_TBL | LOCATION | NA |
| 3 | ou | 1 | No | DEPT_TBL | DEPTID | NA |

## Translating or Performing Functions With Database Values

Access the DN Attribute Function - Directory Interface page.

**Translate Value** Select to replace the database value with the Distinguished Name field value for the selected attribute.

**PeopleCode Function** Select to use the selected database object value as a parameter in a PeopleCode function. The system uses the resulting value as the attribute's distinguished name.

**Don't Transform value** Select to instruct the system to keep the database value as is. This option is the default value for this field.

**Database Value** Enter the database value that you want the system to replace. For example, every time the database value *Vancouver* appears in the Location attribute, the system replaces it with the distinguished name*Van*.

This field is available only when you select Translate Value as the transformation option.

**Distinguished Name** Enter the distinguished name value to replace the database value with.

This field is available only when you select Translate Value as the transformation option.

**PeopleCode Function Name** Enter the PeopleCode function that the system should use to calculate the distinguished name for the selected attribute.

This field is available only when you select PeopleCode Function as the transformation option.

### Setting Up PeopleCode Attribute-Level Functions

When the mapping function accesses the values in the selected field, the field value is passed into a PeopleCode function as a parameter and the output is assigned to the attribute in the directory.

Before you can enter a function on this page in the PeopleCode Function Name field, you must set up the function in the FUNCLIB_EO_DS.DSDYNFUNC FieldFormula.


Copyright © 1992-2006, Oracle. All rights reserved.

To create a function:

1. Open the FUNCLIB_EO_DS.DSDYNFUNC FieldFormula.
2. Add a section in DSDynamicAttrFunc.
3. In the evaluate statement, add the following section for each function that you want to add (*FuncX* is equal to your function name):

```
When = 'FuncX'
   FuncX(&AttrIn, &AttrRT);
   Break;
```

4. Define a DSDynamicAttrFunc PeopleCode function.

   The parameter list must contain two parameters, an attribute type string input and an attribute type string output.

### PeopleCode Function Example

The following example displays the setup for functions FuncX, FuncY, and FuncZ.

DSDYNFUNC (field) | FieldFormula

```
Function FuncX(&AttrIN As string, &AttrRT As string)
End-Function;

Function FuncY(&AttrIN As string, &AttrRT As string)
End-Function;

Function FuncZ(&AttrIN As string, &AttrRT As string)
End-Function;

Function DSDynamicAttrFunc(&FuncName As string, &AttrIN As string, &AttrRT As string)
   Evaluate &FuncName
   When = "FuncX"
      FuncX(&AttrIN, &AttrRT);
      Break;
   When = "FuncY"
      FuncY(&AttrIN, &AttrRT);
      Break;
   When = "FuncZ"
      FuncZ(&AttrIN, &AttrRT);
      Break;
   When-Other
      Break;
   End-Evaluate;

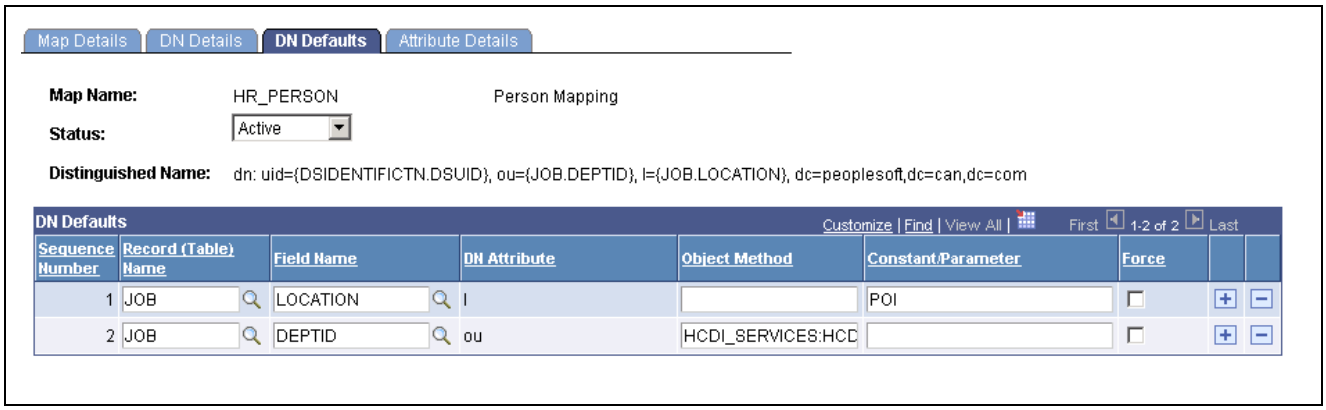
End-Function;
```

Setup functions on the FUNCLIB_EO_DS.DSDYNFUNC FieldFormula

## Specifying Distinguished Name Defaults

Access the DN Defaults page.

Copyright © 1992-2006, Oracle. All rights reserved. 




DN Defaults page

On the DN Defaults page, define a constant value or a PeopleCode method that returns a value to populate blank values when you are building Directory Information Trees. This page enables you to define defaults for any Record.Field value that is left blank in the data rowset of the message that is used to populate the map. For example, you can enter defaults to fill in the blanks left by the lack of Department/Location data for the Persons of Interest constructed by the Directory Interface.

In the preceding screen shot, the value for JOB.DEPTID defaults to a method called HCDI_SERVICES:HCDIUtilities.DeptID. This method returns a DeptID constant.

---

**Note.** The syntax for the method needs to be fully qualified using the following format: Package_Name:App_Class_Name.Method_Name.

---

**Seq** (sequence number) Displays the sequence number for this attribute.

**Record (Table) Name** Select the record name for the value. This field is required.

**Field Name** Select the field name for the value. This field is required.

**DN Attribute** The name of the Distinguished Name attribute.

**Object Method** Enter the object method you are using to populate the value, if applicable. Leave this field blank if you are using a constant or parameter to populate the value.

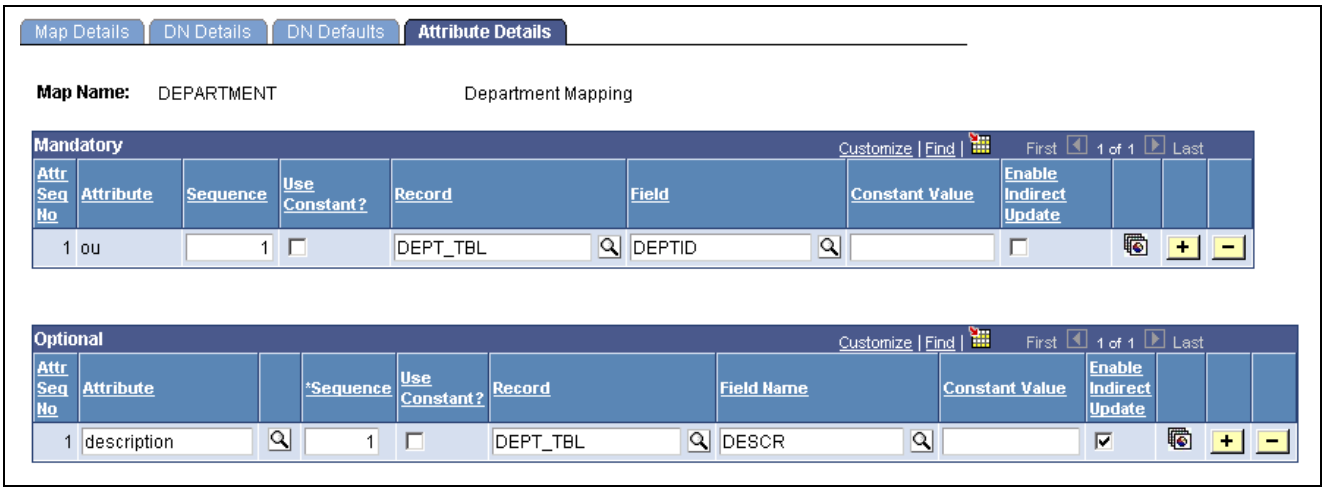
**Constant/Parameter** Enter the values for the constant or the parameter, if applicable. Leave this field blank if you are using an object method to populate the value.

**Force** Select this check box to overwrite the Record.Field values at runtime, even if the values exist.

# Mapping Data to Directory Object Class Attributes

Access the Attribute Details page.


Copyright © 1992-2006, Oracle. All rights reserved.




Attribute Details page

On the Attribute Details page, associate the fields contained in the message that you selected on the Map Details page with the attributes that provide more detail about an entry. Some attributes are mandatory (an object class's mandatory attributes are defined in the directory schema) and must be mapped to either a constant value or record or field. For the department example, you would map PeopleSoft records and fields to the mandatory attributes (such as DeptID), and you could add additional attributes that would give you more information about the object class, such as description.

---

**Note.** The system does not update related-display field values unless the source field is also mapped. If the source field is not mapped, the audit process still indicates and enables you to update any discrepancies. For example, when you map to an employee's job code, the directory entry also includes the job code description. If you change the job code description on the Job Code component, the system updates the related-display description field on the employee's Job Data page, but it does not update to the directory, because it is not included in the mapping.

---

***Warning!*** The fields that you map to mandatory attributes must contain data or the mapping will fail. You can guarantee that there will be data in the fields by mapping mandatory attributes to required fields.

---

**Attr Seq No** (attribute sequence number) — Displays the attribute sequence number assigned to this attribute.

**Attribute** — In the Mandatory scroll area, the system displays the mandatory attributes for this object class.

In the Optional scroll area, select optional attributes.

**Seq** (sequence number) — Enter a sequence number for this attribute. Some directory attribute values are made up of multiple values. The attribute sequence number distinguishes between the different attribute values and indicates to PeopleSoft Directory Interface the order in which the PeopleSoft and constant values should be assigned to the attribute.

**Ind Upd** (indirect update) — Select if the field that you selected is used as an attribute in the directory outside of this mapping and you want it to be updated when this field is updated. The system only updates attributes in entries at lower levels on the directory information tree than this entry.

Copyright © 1992-2006, Oracle. All rights reserved.

## Locating Delivered Messages

Your PeopleSoft application that supports the PeopleSoft Directory Interface delivers a set of messages to be used share information with your directory service.

**Note.** If you have upgraded from a previous PeopleTools 8.48x release, the upgrade program creates service operations for these messages. The service operation names and message names are the same.

For information on this delivered data and how it works in conjunction with the PeopleSoft Directory Interface, see your PeopleSoft application documentation.

# (Optional) Setting Up Entry Membership Rules

This section discusses how to:

- Create entry definitions.
- Specify entry membership rules.

Entry membership rules enable you to modify a directory entry, such as a group, based on criteria stored in the PeopleSoft database. This feature provides a method to match any type of directory entry to rules that are meaningful in PeopleSoft. You can use membership rules to create any type of logical grouping in the directory. The groupings are not restricted to security purposes.

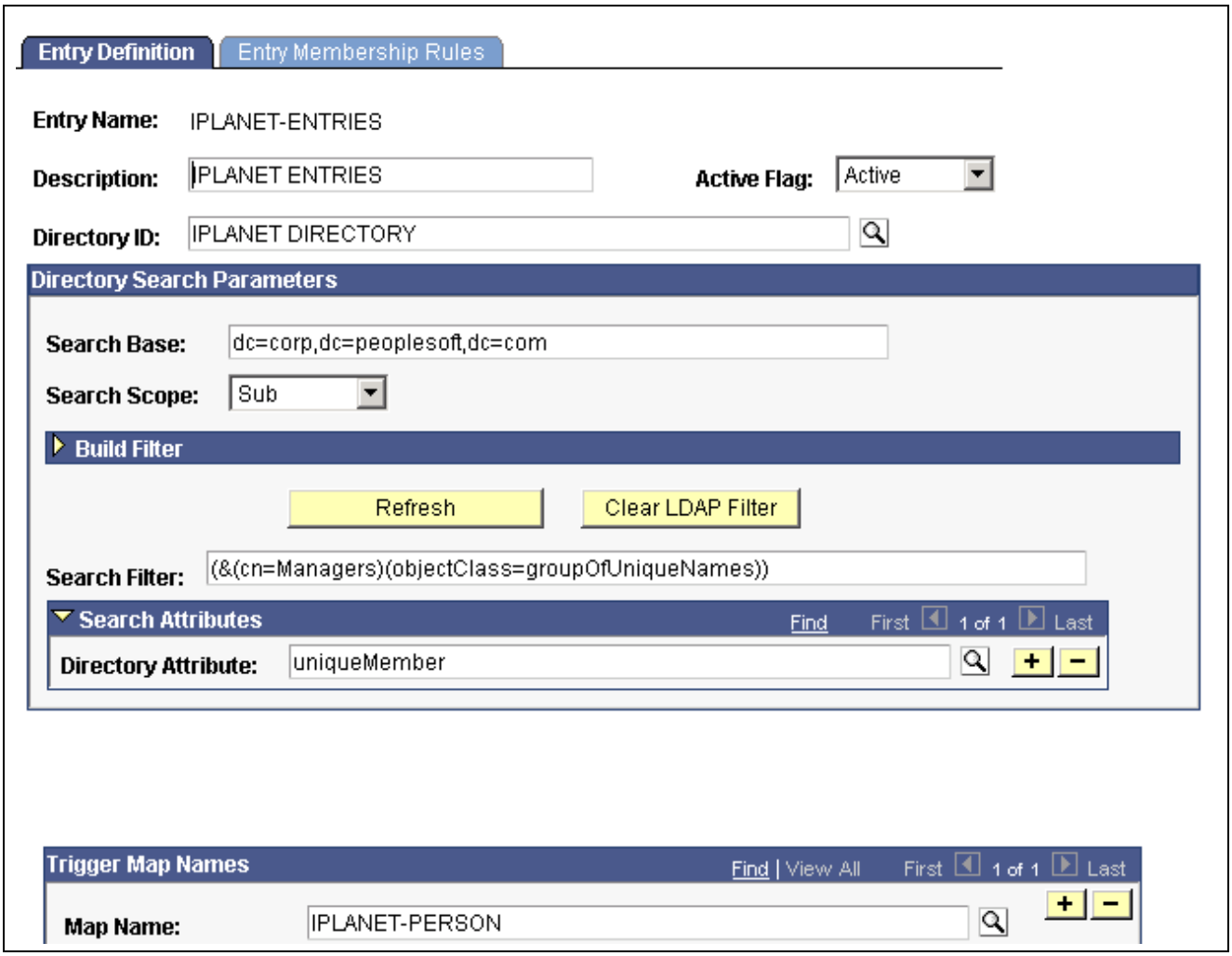
## Pages Used to Set Up Entry Membership Rules

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Entry Definition | EO_DSCONTAINERDEFN | Enterprise Components, Directory Interface, Membership Rules, Entry Rules, Entry Definition | Create a directory entry definition. |
| Entry Membership Rules | EO_DSSECRULE | Enterprise Components, Directory Interface, Membership Rules, Entry Rules, Entry Membership Rules | Establish entry membership rules. |

## Creating Entry Definitions

Access the Entry Definition page.

 Copyright © 1992-2006, Oracle. All rights reserved.




Entry Definition page

**Entry Name** Displays the entry name that you entered on the search page. The system uses this value for the entry name throughout the application, so it must be the name of an existing entry in the external directory. PeopleSoft assumes that the name is unique in the directory.

**Active Flag** Select to enable rules. Rules that aren't active do not run.

## Directory Search Parameters

**Search Base** Enter the distinguished name of the base under which this entry will be located in the directory. The application performs an LDAP search to retrieve the distinguished name of the entry using this field as the base.

**Search Scope** Select from:

*Base:* The query searches only the value in the Search Base field.

*One:* The query searches only the entries one level down from the value in the Search Base field.

*Sub:* The query searches the value in the Search Base field and all entries beneath it.

Copyright © 1992-2006, Oracle. All rights reserved.

### Build Filter

**( )** Select the check boxes below the parentheses to group expressions. You can group more than one line together using the check box on the left for the first line and the check box on the right for the last line.

**Attribute** Enter the name of the attribute that will store the members of the entry in the external directory. It is typically set to *member,* but the attribute name could be anything that you choose.

**Operation** Assign an operator to your rule such as <, <=, <>, =, >, or >=.

**Value** Assign a value to the attribute in your rule.

**And/Or** To add another line to your rule, select *AND* or *OR* depending on your rule logic. Select *END* to signify the end of the search. Select *NONE* if you are not using this kind of filter.

**Refresh** After you make changes using the Build Filter options, click this button to update the Search Filter edit box to reflect the changes.

**Clear LDAP Filter** Click this button to delete all values from the Search Filter edit box and the Build Filter selections.

**Search Filter** Displays the filter that the system applies to the search for the distinguished name of the defined entry. Typically displays the directory object class of the entry in the form "objectclass = GroupOfUniqueNames", for example. This indicates what type of entry to search. To retrieve the correct entry distinguished names, the system adds the name of the entry to the search filter at runtime. The name retrieved by the LDAP search using this filter is tied to the rules defined in the Entry Membership Rules page. When these rules run, the employee that the system is currently processing is either added to or deleted from the distinguished name retrieved by the search.

### Search Attributes

**Directory Attribute** Select the attribute of the entry being defined that will contain all the members of this entry. This attribute must be valid for the current entry in the directory. The employees that satisfy the entry membership rules of this entry are added under this entry as a new value of this attribute. Because of this, there will be as many attribute values as there are employees satisfying the entry membership rules. If this field is left blank, the application uses *member* as a default attribute name.

### Trigger Message Names

**Map Names** Select the names of the maps to associate with the entry definition. Besides being a security feature, this also improves performance at runtime, because only applicable rules are evaluated.

---

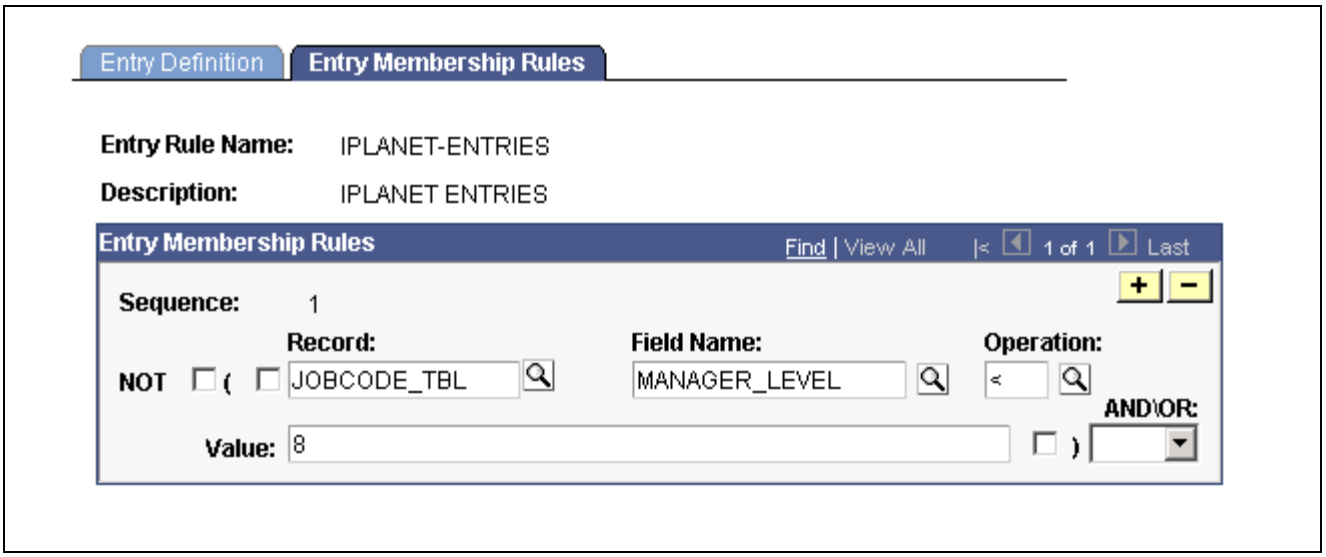
**Note.** Run the directory audit if an entry rule has changed or if you want to initialize the directory entries.

---

## Specifying Entry Membership Rules

Access the Entry Membership Rules page.

 Copyright © 1992-2006, Oracle. All rights reserved.




Entry Membership Rules page

## Entry Membership Rules

**Sequence** Displays the sequence of a rule within a rule set. The sequence becomes significant when you enter more than one rule.

**NOT** Select to negate the rule that you enter. This is similar to using the symbol *!* to reverse the truth value of an operand.

**( )** Select the check boxes to add parentheses around your rule. You can group more than one line together using the check box on the left for the first line and the check box on the right for the last line.

**Record** and **Field Name** Enter the name of the PeopleSoft record and field containing the information to be tested.

**Operation** Enter the appropriate operator, such as < , <= , <> , = , > , or >=.

**Value** Enter the value that the employee's data needs to be tested on. This can be any value of the same type as the field used in the rule, such as String, number, date, and so on.

**AND/OR** To add another line to your rule, select *AND* or *OR* depending on your rule logic. Select *END* to signify the end of the search. Select *NONE* if you are not using this kind of filter.

The entry rules are logical expressions that can be either true or false. They are composed of filters on database objects associated by logical operators. Rules have the following form:

```
[NOT] [ ( ]  Record . Field   operatorConstant [ ) ] [AND/OR]
```

The symbols between square brackets are optional. The operator can be <, <=, <>, =, >, or >=. A rule set is composed of single rules joined by AND or OR Boolean operators if necessary. The following example shows a series of single rules joined to make one compound rule.

```
( JOB.LOCATION = 'KC004' AND [1]
JOB.COMPRATE > 15000 ) OR [2]
NOT JOB.DEPTID = 'GBIY004'[3]
```

Copyright © 1992-2006, Oracle. All rights reserved.

**Note.** There are no limits to the number of rules used within a rule set.

# Loading Data into the Directory

This section provides an overview of loading the directory and discusses how to load the directory with PeopleSoft data.

## Understanding Directory Load Behavior

Use the Directory Load process when there is no existing data in the directory. The process overwrites any data in the directory.

If you have data in your directory, use the Directory Audit process instead of the Directory Load process. The audit process compares the PeopleSoft data to your existing directory data and enables you to review and resolve any possible conflicts.

**Note.** For HRMS customers only, there is an alternative process named DSMAPINPUT FullSync that you can use in place of the Directory Load process. This new process does not replace the Directory Load process; it is provided as an alternative to load the data if performance becomes an issue.

See *PeopleSoft Enterprise HRMS Application Fundamentals PeopleBook*

## Pages Used to Load PeopleSoft Data into the Directory

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Directory Load | EO_RUNCTL_DS_LOAD | Enterprise Components, Directory Interface, Load Directory | Run the Directory Load process. |

## Loading the Directory with PeopleSoft Data

Access the Directory Load page.

 Copyright © 1992-2006, Oracle. All rights reserved.




Directory Load page

**LDIF File** Select to have the process send the data to an LDIF file for you to load in the directory.

**Direct Update** Select to have the process directly update the directory.

**Run** Click to run the process using PeopleSoft Process Scheduler.

Copyright © 1992-2006, Oracle. All rights reserved.

 Copyright © 1992-2006, Oracle. All rights reserved.

# CHAPTER 3

# Reviewing Directory Data and Generating Reports

This chapter discusses how to:

- Review LDAP directory data.
- View PeopleSoft Directory Interface reports.
- Manage transaction audit history.

## Reviewing LDAP Directory Data

This section discusses how to:

- Run a directory audit.
- Run a directory search.

### Pages Used to Review Directory Data

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Directory Audit | EO_RUNCTL_DS_AUDIT | Enterprise Components, Directory Interface, Run Directory Audit | Run the Directory Audit process. |
| Directory Search | EO_DSSRCHDIRECTORY | Enterprise Components, Directory Interface, Search Directory | Define search parameters to query the directory. The page saves the search parameters for future use. |

### Running a Directory Audit

Access the Directory Audit page.

Use the Directory Audit process to ensure that your directory database has the same data as your PeopleSoft database. The Directory Audit process compares the data in the directory to the data in the PeopleSoft database identified in the selected map and creates an LDAP Data Interchange Format (LDIF) file containing any discrepancies using PeopleSoft as the authority. You can then use the LDIF file to update the directory.

**Note.** For HRMS customers only, there is an alternative process named DSMAPINPUT FullSync Audit that you can use in place of the Directory Audit process. This new process does not replace the Directory Audit process; it is provided as an alternative to audit the data if performance becomes an issue.

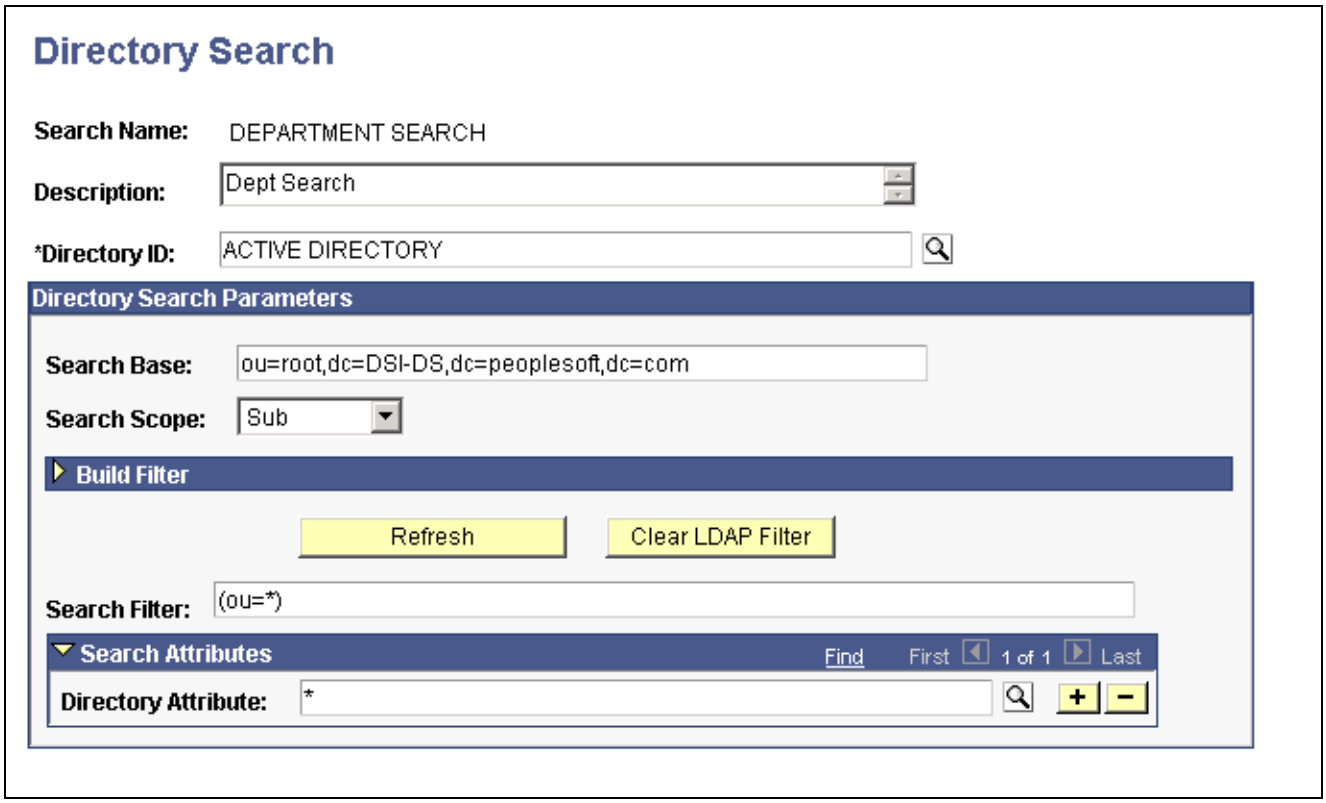
See *PeopleSoft Enterprise HRMS Application Fundamentals PeopleBook*

Copyright © 1992-2006, Oracle. All rights reserved.

**Map Name** Select the name of the map that the audit should be run against.

# Running a Directory Search

Access the Directory Search page.




Directory Search page

Use the Directory Search page to define search parameters to query the directory and view the results. Search results are displayed on the Search Results page as they appear in the directory.

**Search Name** Enter the search name. The system saves the search parameters that you enter on this page and stores them under this name for future use.

## Directory Search Parameters

**Search Base** Select the directory entry that is the search base for this search. The search base is the entry in the directory information tree at which the search begins querying.

**Search Scope** Select from:

*Base:* The query searches only the value in the Search Base field.

*One:* The query searches only the entries one level down from the value in the Search Base field.

*Sub:* The query searches the value in the Search Base field and all entries beneath it.


Copyright © 1992-2006, Oracle. All rights reserved.

### Build Filter

Use the fields in the Build Filter group box to create an attribute-specific filter. For example, if you want data on a single person, enter the attribute name *Person,* the operation *=,* and enter the person's name in the Value field. You can construct multiple filters.

**( )** Use parentheses to separate filter statements.

**Attribute Name** Enter the name of the attribute whose data you want to filter.

**Operation** Select an operator to determine how to filter the data.

**Value** Select a value to compare against when filtering.

**And/Or** Select from:

- *And:* A search must meet the criteria of multiple statements.
- *END:* This value marks the end of a query. This value applies parentheses appropriately.
- *NoOp:* No operator is used.
- *OR:* A search must meet the criteria of one of multiple statements.

**Search Filter** You can narrow the search (for example, instruct the system to search for all attributes but one) by entering a search filter. Enter the search filter using standard LDAP protocol.

You can specify one or more attributes to search for. To search all attributes, enter an asterisk ( * ).

For more information about the LDAP protocol, see your directory documentation.

### Search Attributes

**Directory Attribute** Select the attributes to search for.

Leave this field blank to search all attributes.

# Viewing PeopleSoft Directory Interface Reports

This section discusses how to:

- View the directory audit report.
- View the business interlink status report.

Copyright © 1992-2006, Oracle. All rights reserved.

# Pages Used to View PeopleSoft Directory Interface Reports

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Directory Audit Report | EO_RUN_DS_AUD_RPT | Enterprise Components, Directory Interface, Audit Report | Generate the Directory Audit report. |
| Directory BI Status Report | EO_RUN_DS_BI_RPT | Enterprise Components, Directory Interface, Bus Interlink Status Report | Generate the Directory BI Status report. |

## Viewing the Directory Audit Report

Access the Directory Audit Report page.




Directory Audit Report page

The Directory Audit SQR report (EO_DS001) locates and reports discrepancies between the PeopleSoft database and your directory. Before you can run the Directory Audit report, run the Directory Audit process. The Directory Audit process populates a comparison record containing the data that differs between the PeopleSoft database and the directory and creates an LDIF file with this data that can be used to update the directory. The Directory Audit report is based on this record, so you can verify what will be updated in the directory if you apply the LDIF file.

The report generates the following errors:

1. The distinguished name is not found in the directory.
2. The distinguished name is not found in PeopleSoft.
3. The attribute is in PeopleSoft but not in the directory.
4. The attribute is in the directory but not in PeopleSoft.
5. The value is in PeopleSoft but not in the directory.
6. The value is in the directory but not in PeopleSoft.

## Viewing the Business Interlink Status Report

Access the Directory BI Status Report page.


Copyright © 1992-2006, Oracle. All rights reserved.




Directory BI Status Report page

If you selected an output type of Business Interlinks when setting up maps to associate PeopleSoft fields to directory attributes, the system uses PeopleSoft Business Interlinks to modify the directory. If errors are produced as a result of the interlinks, the system writes the errors to an error record. The Business Interlink Status SQR report (EO_DS002) retrieves and presents the data contained in this error record.

**Delete History Error Rows for MAP** Select to delete historical error rows for this map after reporting them. The PS_EO_BILOAD_ERR record retains error data for this map until you run the report with this check box selected.

**Run** Click Run to run the report using PeopleSoft Process Scheduler.

### See Also

*Enterprise PeopleTools 8.48 PeopleBook: PeopleSoft Process Scheduler*

# Managing Transaction Audit History

This section discusses how to:

- Run a transaction history report.
- Search for a transaction history.
- Purge transaction history.

Copyright © 1992-2006, Oracle. All rights reserved.

# Pages Used to Manage Transaction Audit History

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Transaction History Report | EODS_RUN_AUDIT | Enterprise Components, Directory Interface, Audit History, Run Report | Generate a transaction history report. |
| Transaction History Inquiry | EODS_AUDIT_HIS_IN | Enterprise Components, Directory Interface, Audit History, Transaction Inquiry | Search for a transaction history. |
| Purge Transaction History | EODS_RUN_ADT_PURGE | Enterprise Components, Directory Interface, Audit History, Purge Transactions | Purge audit history data. |

# Running a Transaction History Report

Access the Transaction History Report page.




Transaction History Report page

**Map Name** Select a Directory Entry Map name.

**User ID** Select the user ID.

**Status** Select the status of the transaction. Valid values are Successful or No Success.

**Output Type** Select the type of output. Valid values are File Format or Business Interlink.

**Transaction Type** Select a type of transaction. Valid values are Add, Delete, and Update.

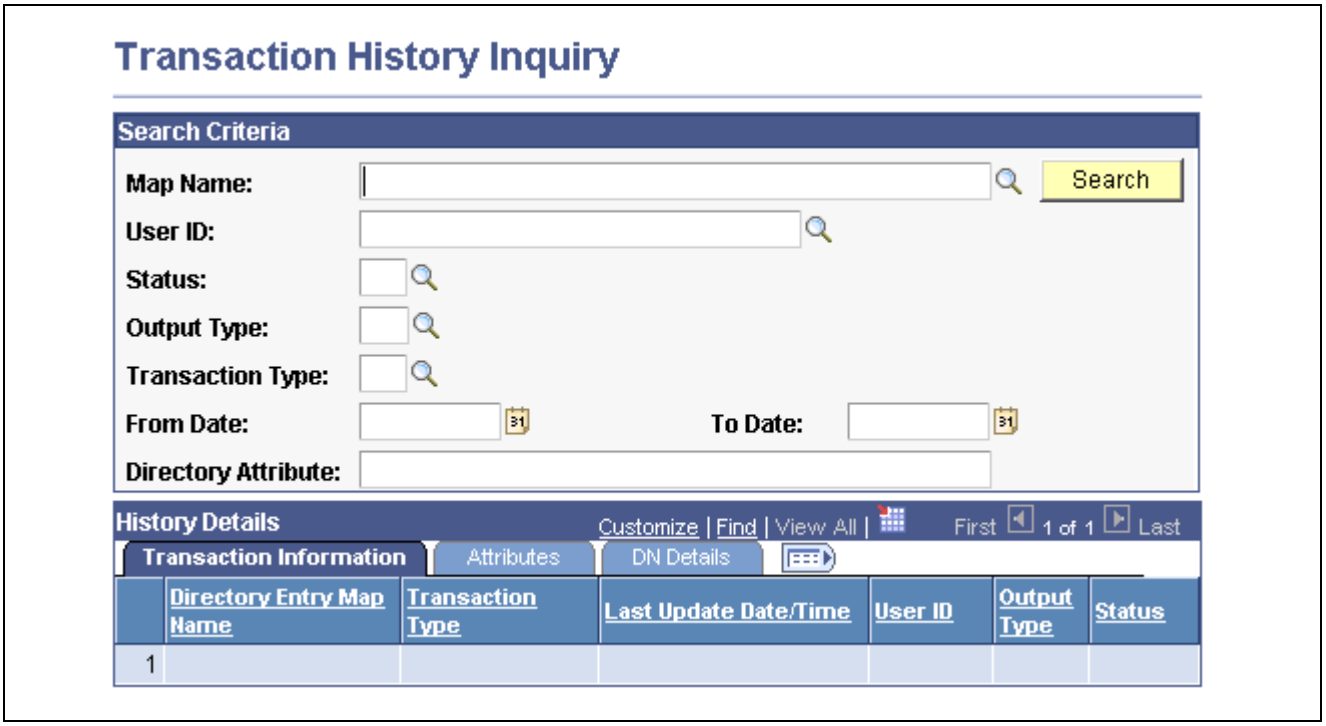
**Directory Attribute** Select the attribute of the entry being defined that will contain all the members of this entry. This attribute must be valid for the current entry in the directory. The employees that satisfy the entry membership rules of this entry are added under this entry as a new value of this attribute. Because of this, there will be as many attribute values as there are employees satisfying the entry


Copyright © 1992-2006, Oracle. All rights reserved.

membership rules. If this field is left blank, the application uses *member* as a default attribute name.

# Searching for a Transaction History

Access the Transaction History Inquiry page.




Transaction History Inquiry page

Use the Transaction History Inquiry page to run a query that displays in the browser rather than create a report that runs using Process Scheduler.

**Map Name** Select a Directory Entry Map name.

**User ID** Select the user ID.

**Status** Select the status of the transaction. Valid values are Successful or No Success.

**Output Type** Select the type of output. Valid values are File Format or Business Interlink.

**Transaction Type** Select a type of transaction. Valid values are Add, Delete, and Update.

**Directory Attribute** Select the attribute of the entry being defined that will contain all the members of this entry. This attribute must be valid for the current entry in the directory. The employees that satisfy the entry membership rules of this entry are added under this entry as a new value of this attribute. Because of this, there will be as many attribute values as there are employees satisfying the entry membership rules. If this field is left blank, the application uses *member* as a default attribute name.

## Transaction Information Tab

The Transaction Information tab displays information about the transaction including the map name, the status, output type, and transaction type.

Copyright © 1992-2006, Oracle. All rights reserved.

### Attributes Tab

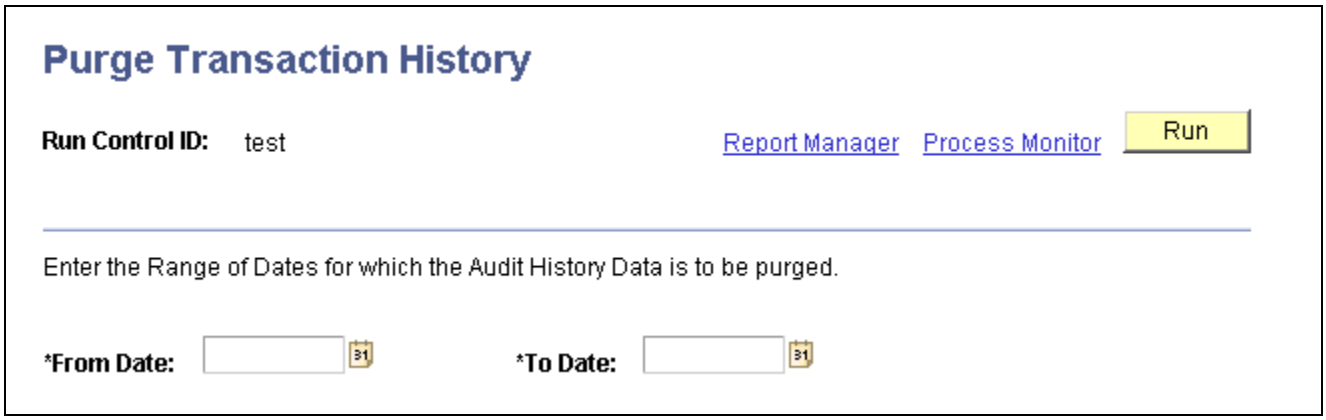
The Attributes tab displays information about the transaction including what directory audit action was performed, the type of directory attribute, and the attribute value.

### DN Details Tab

The DN Details tab displays the distinguished name of the transaction.

## Purging Transaction History

Access the Purge Transaction History page.




Purge Transaction History page

Use the Purge Transaction History page to purge audit history data based on the date range that you enter in the run control page.


Copyright © 1992-2006, Oracle. All rights reserved.

# PART 3

# Common Objects and Components

**Chapter 4**
**Working With Currencies and Market Rates**

**Chapter 5**
**Using Datasets**

**Chapter 6**
**Using Interactive Reports**

**Chapter 7**
**Setting Up the Credit Card Interface**

# CHAPTER 4

# Working With Currencies and Market Rates

This chapter provides an overview of currencies and market rates and discusses how to:

- Define currencies.
- Define currency quotation methods.
- Define market rates.
- Calculate cross and reciprocal rates.
- Use the Currency Exchange Calculator.

## Understanding Currencies and Market Rates

PeopleSoft offers a core set of objects (fields, tables, work records, pages, and PeopleCode functions) and a recommended set of standard techniques and formulas to support a common approach to converting currency throughout PeopleSoft applications, and for defining and storing market rates. Market rate is a generic term for a currency exchange rate, an interest rate, or a future rate, for example.

In this section, we discuss:

- Currency and market rate tables.
- Conversion factor fields and the visual rate.
- Application-specific requirements for currency conversion.

### Understanding Currency and Market Rate Tables

The following tables store currency and market rate data:

- CURRENCY_CD_TBL

  Stores currency code data.
- CURR_QUOTE_TBL

  Stores currency quotation method data.
- RT_INDEX_TBL

  Stores market rate index data.
- RT_TYPE_TBL

  Stores rate type data.
- RT_RATE_TBL

Copyright © 1992-2006, Oracle. All rights reserved.

Stores market rate data.

## Understanding Triangulation

Triangulation is the process by which a conversion between two currencies takes place by way of a third reference currency. This process may be used in hyperinflationary environments, where all conversions to the local currency are done by way of a stronger, more stable currency. This process may also be used when a country is undergoing a currency revaluation.

To support triangulation, PeopleSoft provides a means to define that you want a currency pair to triangulate through a fixed reference currency. The actual conversion process is done in a two-step procedure in which the from-currency amount is first converted to the reference currency and then to the destination currency, using the appropriate exchange rates. Supporting triangulation also affects the user interface, as there are now two or possibly three exchange rates that are relevant to the conversion. When viewing a triangulated conversion at a detailed level, users access three visual rates:

- A rate for converting the from-currency to the reference currency.
- A rate for converting the reference currency to the to-currency.
- A cross rate indicating the rate that would be required to convert the from-currency directly into the to-currency.

The cross rate in a triangulated conversion is not typically maintained directly. The system enables you to maintain those non-triangulated rates that are components of the triangulated rate, then run a process to generate the triangulated exchange rate. However, you can override the cross rate, which causes one of the other exchange rate values to be recalculated to synchronize it with the overridden cross rate.

For example, suppose an implementation was using triangulation to convert from USD to FRF. You would directly maintain the visual rate from the USD to euros (1.25 in the example table) and rate from euros to FRF (6.8 in the example table). You could then run the EOP_RATECALC Application Engine process to derive the triangulated rate for converting from USD to FRF. The results are shown in the following table:

| Currency Pair | Quote Method | Quote Units | Primary Visual Rate | RATE_MULT | RATE_DIV |
|---|---|---|---|---|---|
| USD to Euro | Indirect | 1 | 1.25 | 1 | 1.25 |
| Euro to FRF | Direct | 1 | 6.8 | 6.8 | 1 |
| USD to FRF | Direct /Triangulate /Euro | 1 | 5.44 | 6.8 | 1.25 |

For performing the actual conversion, applications interpret the visual rates into RATE_MULT and RATE_DIV values based on the quotation method for the exchange, then use the RATE_MULT and RATE_DIV values stored in the Market Rates Data table in the currency conversion formula, either by accessing the values directly or by calling the ConvertCurrency PeopleCode function.

**Note.** For information on how a specific application supports maintenance of triangulated exchange rates, see the documentation for that application.


Copyright © 1992-2006, Oracle. All rights reserved.

### See Also

Chapter 4, "Working With Currencies and Market Rates," Calculating Cross, Triangulated, and Reciprocal Rates, page 59

## Understanding Conversion Factor Fields and the Visual Rate

Support for both direct and indirect currency quotations creates a potential for complex currency conversion formulas in applications. To avoid excess conditional logic in the conversion formula, PeopleSoft provides two fields to store the conversion factor, RATE_MULT and RATE_DIV. The rate that you enter is called the *visual rate.* This visual rate is generally stored in either RATE_MULT or RATE_DIV, based on the quote method. The quote units are stored in whichever field does not contain the visual rate. As a result, the formula for currency conversion remains consistent:

(from-currency / RATE_DIV) × RATE_MULT = to-currency

This formula is also used for currency conversion in PeopleCode programs for online processing, as well as in SQR and COBOL processes.

The following table shows a few basic examples of how visual rates are transformed into RATE_MULT and RATE_DIV, according to the quote method and quote units for the currency pair:

| Currency Pair | Quote Method | Quote Units | Primary Visual Rate | RATE_MULT | RATE_DIV |
|---|---|---|---|---|---|
| USD to GBP | Indirect | 1 | 1.6 | 1 | 1.6 |
| GBP to USD | Direct | 1 | 1.6 | 1.6 | 1 |
| DEM to CHF | Indirect | 100 | 119.335 | 100 | 119.335 |
| CHF to DEM | Direct | 100 | 119.335 | 119.335 | 100 |
| USD to Euro | Indirect | 1 | 1.25 | 1 | 1.25 |
| Euro to FRF | Direct | 1 | 6.8 | 6.8 | 1 |
| USD to FRF | Direct /Triangulate /Euro | 1 | 5.44 | 6.8 | 1.25 |
| FRF to Euro | Indirect | 1 | 6.8 | 1 | 6.8 |
| Euro to USD | Direct | 1 | 1.25 | 1.25 | 1 |
| FRF to USD | Indirect /Triangulate /Euro | 1 | 5.44 | 1.25 | 6.8 |

Copyright © 1992-2006, Oracle. All rights reserved.

In all cases, the visual rate for a currency pair remains the same, regardless of the direction. This is consistent with business standards. For a direct quoted rate, you multiply by the visual rate; therefore the visual rate goes into RATE_MULT and 1 (or the quote units) goes into RATE_DIV. For an indirect quoted rate, you divide by the visual rate; therefore the visual rate goes into RATE_DIV and 1 (or the quote units) goes into RATE_MULT.

The following examples show indirect quotation, direct quotation with quote units, and triangulation:

100 USD to GBP (indirect) = (100 USD / 1.6) × 1 = 62.50 GBP

1000 CHF to DEM (direct with units) = (1000 CHF / 100) × 119.335 = 1193.35 DEM

100 USD to FRF (triangulate) = (100 USD / 1.25) × 6.8 = 544 FRF

### See Also

Chapter 4, "Working With Currencies and Market Rates," Defining Currency Quotation Methods, page 48

## Understanding Application-Specific Requirements for Currency Conversion

Each application that shows a visual rate on a page must have an application-specific work record to hold the visual rate and the PeopleCode associated with it; this can be an existing work record. The suggested name for the field is VISUAL_RATE. The work record should also have a field to store the original rate for purposes of tolerance checking.

The application also typically provides an application-specific table to store RATE_MULT and RATE_DIV values that are stored on the database.

Application-specific PeopleCode needs to format work record fields and call the common functions in various circumstances, such as RowInit or FieldChange on the currency or visual rate.

### See Also

*Enterprise PeopleTools 8.48 PeopleBook: Global Technology*

*Enterprise PeopleTools 8.48 PeopleBook: PeopleCode Developer's Guide*

# Defining Currencies

In this section, we discuss how to define currencies.

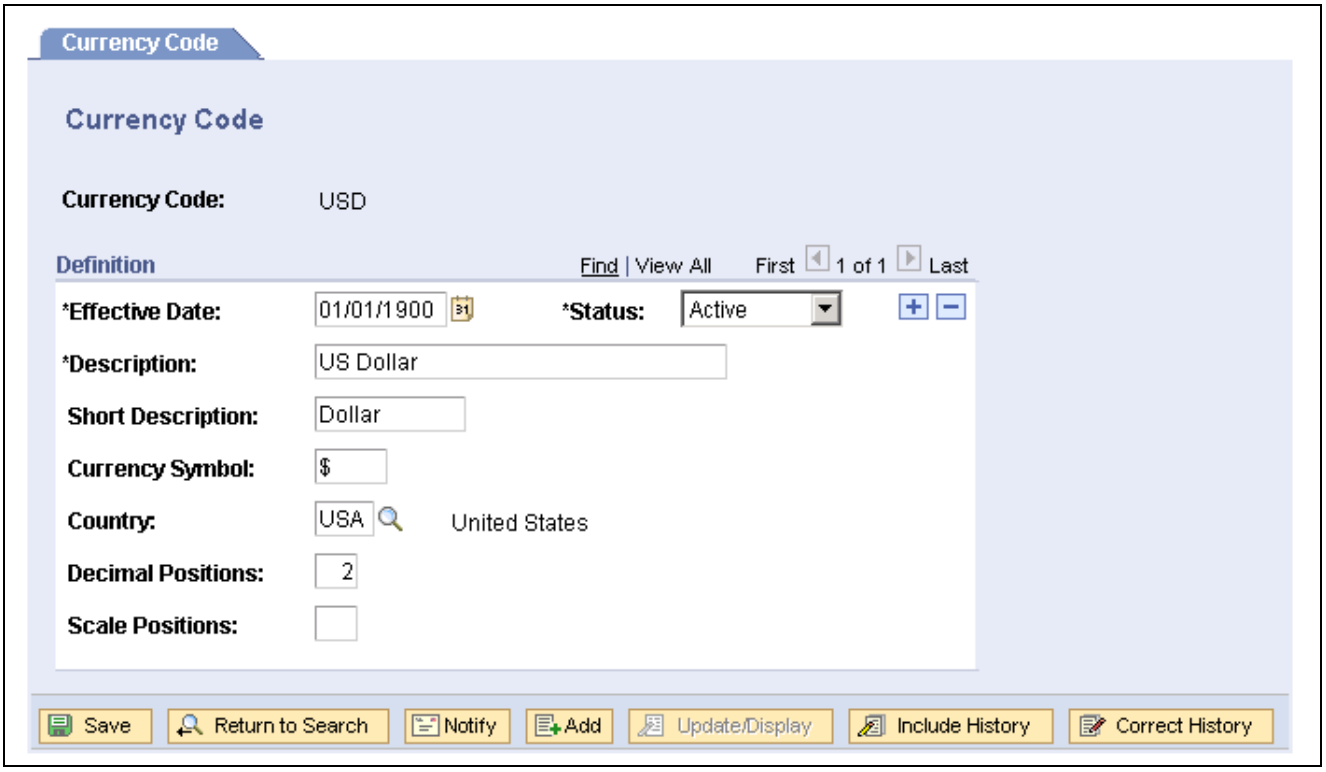
## Page Used to Define Currencies

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Currency Code | CURRENCY_CD_TABLE | Depends on application. | Add and maintain currency codes. These currency codes are used to designate currencies throughout your PeopleSoft system. |

 Copyright © 1992-2006, Oracle. All rights reserved.

# Defining Currency Codes

Access the Currency Code page.




Currency Code page

**Status** Indicate whether the currency code is active or inactive. If you inactivate a currency code that is in use, existing transactions are unaffected. However, the currency code is unavailable for future selections.

Some PeopleSoft applications do not allow you to inactivate a currency code that is in use.

**Currency Symbol** PeopleSoft delivers many currencies with a currency symbol such as $ for Australian dollar (AUD) or £ for British pound (GBP). You can enter new symbols for delivered currencies or for currencies that you might add.

**Country** Select the code for the country from which the currency originates.

**Note.** PeopleSoft delivers fully populated country, state, and province code tables and updates these tables as national boundaries and designations change.

**Decimal Positions** Enter the number of decimal positions that should appear in the notation for the currency. For example, there are two decimal positions for Australian dollars (500.00 AUD), but no decimal positions for Japanese yen (500 JPY).

**Scale Positions** Enter the scale positions you want to round for this currency. This controls how many numbers appear to the left of the decimal when displayed. The data is actually stored with full precision in the database itself.

Copyright © 1992-2006, Oracle. All rights reserved.

For example, if you want all million-dollar amounts displayed as the number of millions without the zeros, enter 6 as your scale position. In this case, 24,000,000 is displayed as 24, but is stored in the database as 24,000,000.

# Defining Currency Quotation Methods

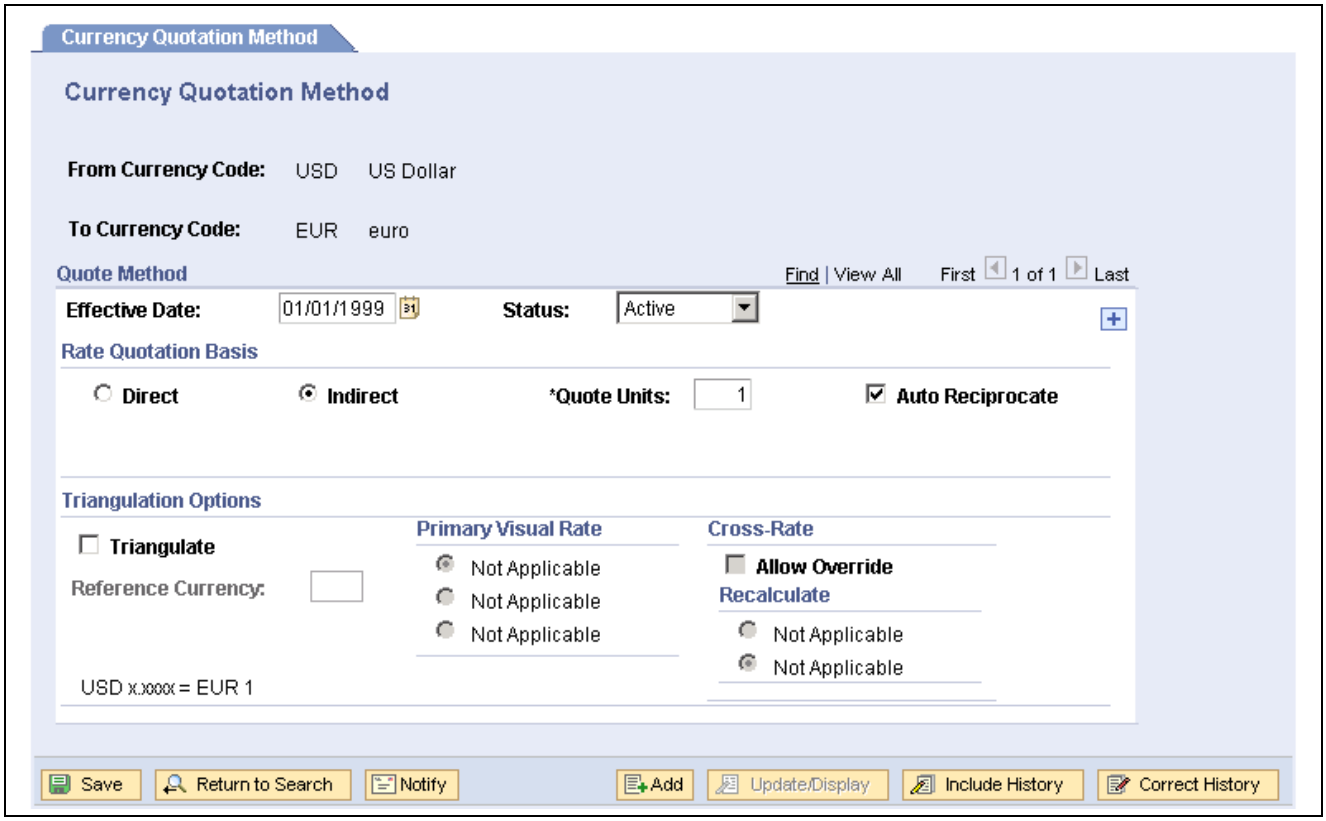
In this section, we discuss how to define currency quotation methods.

## Page Used to Define Currency Quotation Methods

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Currency Quotation Method | CURR_QUOTE_PNL | Depends on application. | Set up and maintain a currency quotation method for each from-currency and to-currency pair. |

## Defining Currency Quotation Methods

Access the Currency Quotation Method page.




Currency Quotation Method page


Copyright © 1992-2006, Oracle. All rights reserved.

A currency quotation method, defined for an exchange rate, stores data that determines how the application interprets a visual rate entered by a user (or multiple visual rates, in the case of triangulated exchange rates) into the RATE_MULT and RATE_DIV values stored on the Market Rate Data table. Conversely, it also determines how the stored RATE_MULT and RATE_DIV values are interpreted into the visual rate displayed to the user.

The quotation method can be direct or indirect, and it can be non-triangulated or a triangulated conversion via a third reference currency. The currency quotation method also determines the quotation units of the from-currency.

See Chapter 4, "Working With Currencies and Market Rates," Understanding Conversion Factor Fields and the Visual Rate, page 45.

It is not necessary to define a currency quotation method for every exchange rate. If, during maintenance of market rates, no quotation method is found for an exchange rate, the page logic assumes these defaults:

- The exchange rate is direct.
- The quotation units are equal to 1.
- The exchange rate is not triangulated.

---

**Note.** This use of default values supports backward compatibility with previous exchange rate data, including calculated reciprocal rates, if your implementation requires them.

---

---

**Note.** You can view the currency quotation method for an exchange rate on the Exchange Rate Detail page while working on the Market Rates page.

---

See Chapter 4, "Working With Currencies and Market Rates," Defining Market Rates, page 50.

**Direct** and **Indirect**

Indicate whether the rates for this currency pair are quoted directly or indirectly. For example, when defining a currency quotation method for USD and FRF:

- Select *Direct* if you want 1 USD to equal x.xxxx FRF.
- Select *Indirect* if you want x.xxxx USD to equal 1 FRF.

Even currency quotation methods for currency pairs that triangulate must be classified as either direct or indirect. In this case, the value is used to display the calculated cross rate.

Support for indirect and direct quotation methods allows applications to eliminate use of calculated reciprocal rates by using a single rate by which you either divide or multiply, depending on whether the conversion method is direct or indirect.

**Quote Units**

Enter a quote unit for the exchange rate, as is common business practice for some currencies. This field can have any value, but is usually a power of 10.

Sometimes called scaling factors, quote units are often used to preserve more decimal precision. For example, the exchange rate between Swiss francs (CHF) and Deutsche marks (DEM) may be stated as 100 CHF = 119.335 DEM instead of 1 CHF = 1.19335 DEM.

**Auto Reciprocate**

Select to automatically create or update the rate for the reciprocal currency pair on the Market Rates page whenever an exchange rate is added or updated.

Copyright © 1992-2006, Oracle. All rights reserved. 

For example, if you create a currency quotation method for USD to EUR. The reciprocal currency quotation method for EUR to USD is automatically created, regardless of this setting.

When you create a rate for USD to EUR on the Market Rates page, the EUR to USD reciprocal rate is automatically created if this Auto Reciprocate option is selected for the currency pair.

If the either rate for the currency pair is updated on the Market Rates page, the reciprocal rate is updated as long as the Auto Reciprocate option is selected for one of the currencies in the pair.

**Triangulate** Select to triangulate conversions between this currency pair using a reference currency.

**Reference Currency** Enter the reference currency for a triangulated conversion.

**Primary Visual Rate** With triangulated currency pairs, there are three exchange rates to consider:

- The rate between the from-currency and the reference currency.
- The rate between the reference currency and the to-currency.
- The calculated cross rate between the from-currency and the to-currency.

Select which of these three rates you want to be the primary visual rate. This is the rate displayed on the primary pages and reports. For online applications, other components of the rate can be viewed and modified on the Exchange Rate Detail page.

**Allow Override** For triangulated currency pairs, select to enable users to override the cross rates on the Market Rates page and Exchange Rate Detail page.

**Recalculate** If the Allow Override option is selected, select to indicate which of the two other rates should be recalculated to bring the triangle back into balance again. Because the triangulated rate is initially a calculated rate, if you allow it to be overridden, the rates used to initially calculate it must be recalculated.

# Defining Market Rates

In this section, we discuss how to:

- Define market rate indexes.
- Define market rate types.
- Create market rate definition.
- Define market rates.
- Access rate definition details.
- Access exchange rate details.

 Copyright © 1992-2006, Oracle. All rights reserved.

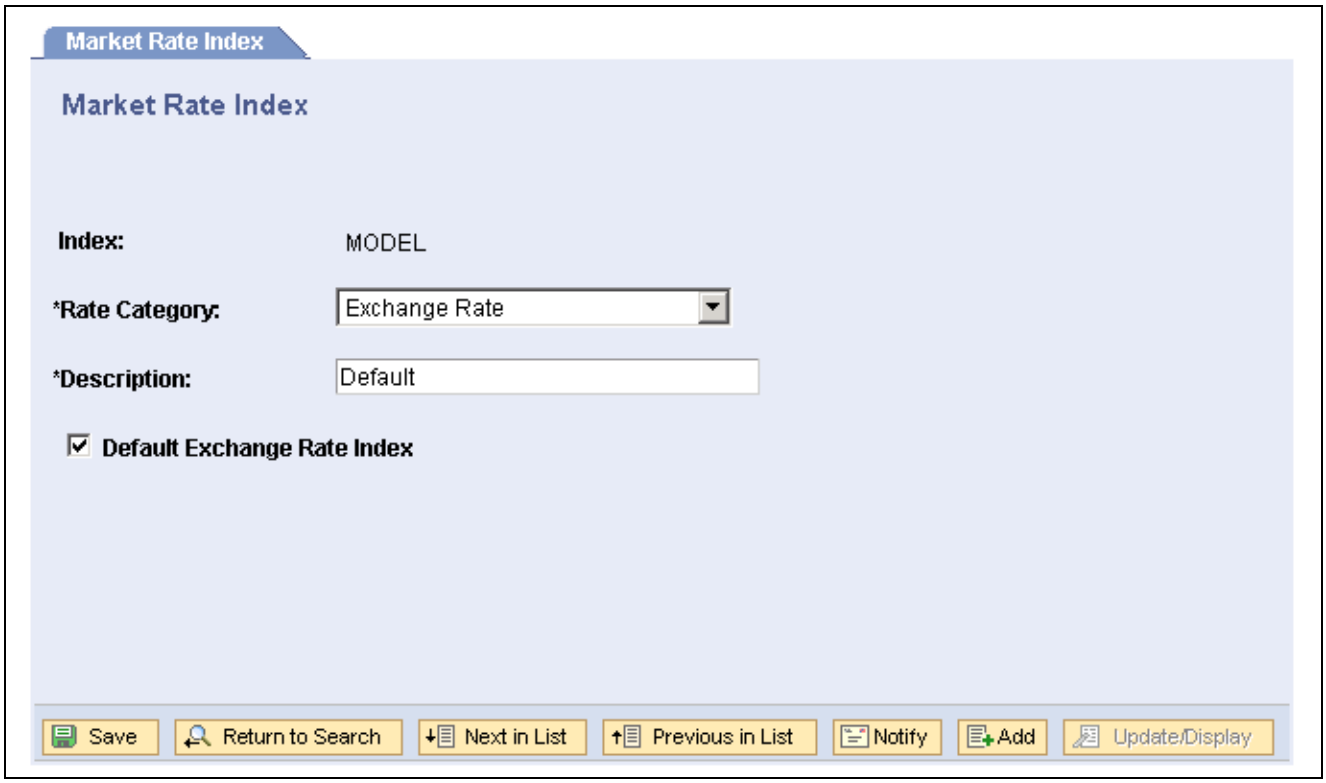
## Pages Used to Define Market Rates

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Market Rate Index | RT_INDEX_TBL | Depends on application. | Create market rate indexes, which provide a means of organizing market rates in the PeopleSoft system. |
| Rate Type | RT_TYPE_TBL | Depends on application. | Define rate types that further categorize market rates. Examples of rate types include current, commercial, floating, average, and historical. |
| Market Rate Definition - Rate Definition | RT_RATE_DEF_TBL | Depends on application. | Define tolerance limits for rates and determine what action occurs if a new rate falls outside the tolerance limit. |
| Market Rates | RT_RATE_PNL | Depends on application. | Maintain and view market rates. The fields available on the page vary depending on the rate category. |
| Rate Definition | RT_RATE_DEF_SEC | Select the Rate Definition link on the Market Rates page. | View market rate definition details, including the maximum variance and error handling definitions specified for the currency pair on the Rate Definition page. |
| Exchange Rate Detail | EXCH_RT_DTL | Click the Exchange Rate Detail button on the Market Rates page. | Access exchange rate detail information. |

## Defining Market Rate Indexes

Access the Market Rate Index page.

Copyright © 1992-2006, Oracle. All rights reserved.




Market Rate Index page

Market rate indexes are stored in the RT_INDEX_TBL table.

**Index** Displays the key term for the highest level of organization for market rates in the application.

**Rate Category** Select a general category for the market rate index, such as *Exchange Rate, Commodity Price,* or *Interest Rate.*

**Default Exchange Rate Index** Select to indicate that the selected market rate index is the default exchange rate index. This field is available only if:

- The Rate Category field is set to *Exchange Rate.*
- No other index is currently defined as the default exchange rate index.

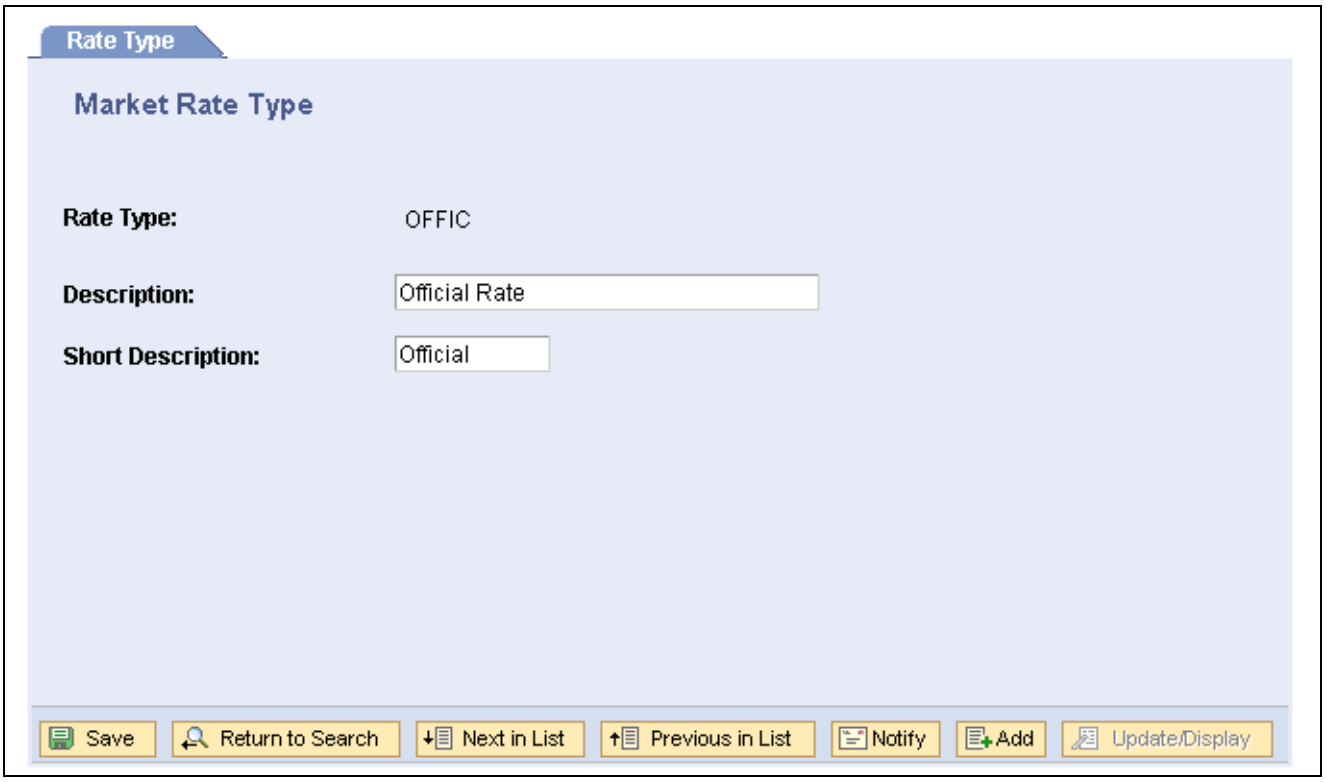
The Market Rates Index page does not ensure that a default market rate index has been defined. However, if no default has been defined, the Market Rate Default view does not return any data.

The Market Rate Definition Default view (RT_DEF_DFLT_VW) selects rows from the Market Rate Definition table that have a term of zero and an index defined as the default exchange rate index.

## Defining Rate Types

Access the Rate Type page.


Copyright © 1992-2006, Oracle. All rights reserved.




Rate Type page

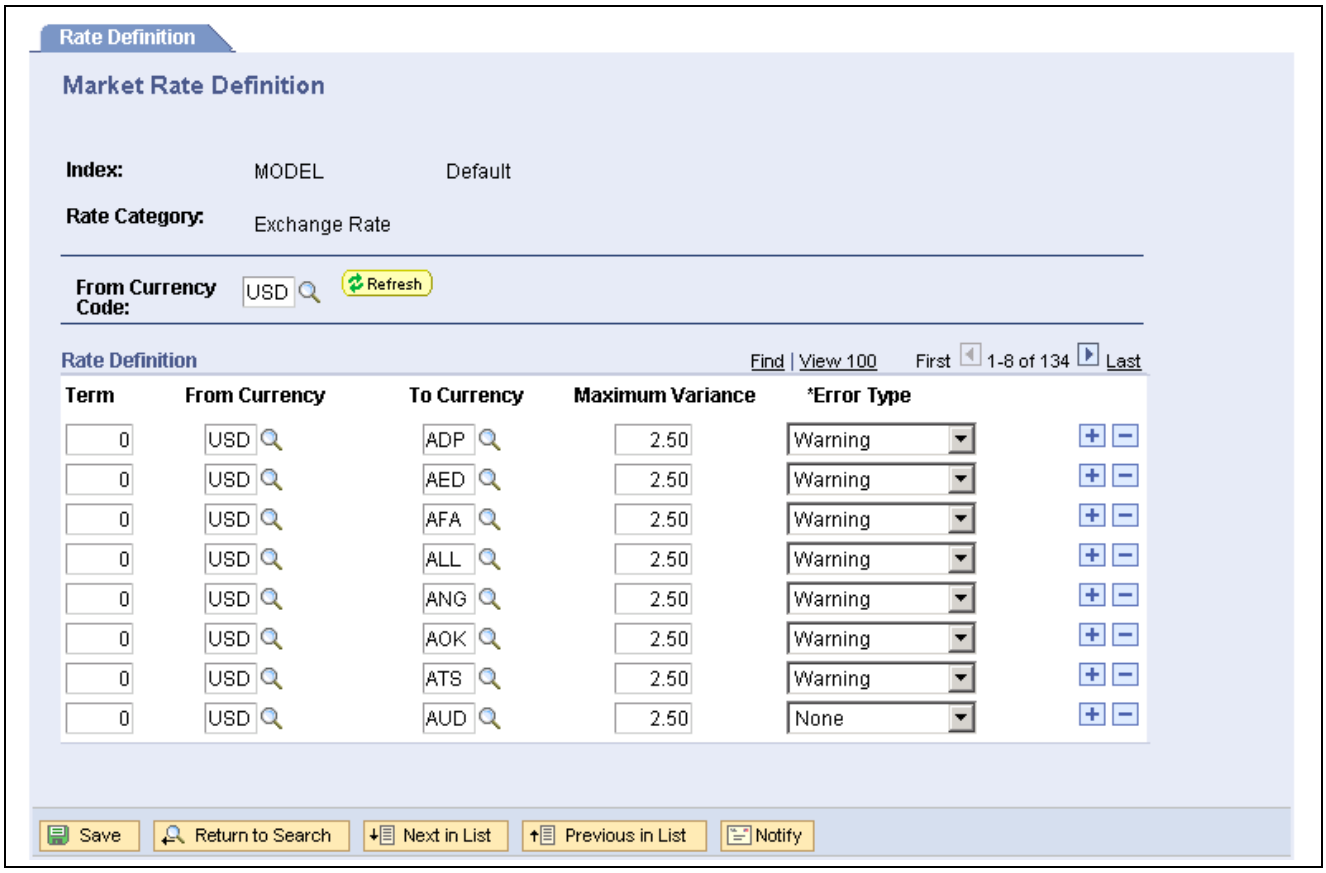
Rate types are stored in the RT_TYPE_TBL edit table. Rate types serve as categories within a market rate index. For example, some common types of exchange rates are official rate, spot rate, and free market rate.

Enter a description and short description to define each market rate type that you use.

## Creating Market Rate Definitions

Access the Market Rate Definition - Rate Definition page.

Copyright © 1992-2006, Oracle. All rights reserved.




Market Rate Definition - Rate Definition page

Market rate definitions specify the valid term, currency, and other appropriate field combinations for market rates. For example, if you have a market rate definition for an exchange rate with a term of 30, a from-currency of CHF, and a to-currency of USD, you can enter a rate using this combination on the Market Rates page.

If you have not created a market rate definition on this page when you create the a market rate on the Market Rates page, the system automatically creates one for you using the default values of 2.5 percent maximum variance and warning message processing.

It is common for applications to support tolerance checking (against user-specified tolerances) in all places where exchange rates can be entered or overridden. With the introduction of indirect quotation methods and quote units, tolerance checking is even more critical to ensure data entry accuracy.

---

**Note.** The information you see on this page depends on the selected market rate index. For example, if you select an index associated with a rate category of *Interest Rate,* fields on this page display interest-related data.

---

**From Currency Code** Enter the from-currency code with which you want to populate all From Currency fields on the page.

**Refresh** Click to populate the From Currency field with the currency you selected in the From Currency Code field.

**Term** Enter the desired term expressed in days. A zero term indicates that the spot rate = zero term. Only PeopleSoft Treasury uses non-zero terms; all other applications must use a zero term for spot rate.

**From Currency** In addition to using the From Currency Code field to populate all From Currency field on this page, you can also manually enter the appropriate


Copyright © 1992-2006, Oracle. All rights reserved.

from-currency. This value is used with its associated To Currency field value as part of an exchange rate pair. When you use triangulation, include a definition row for each of the currency pairs involved in the triangulation.

**To Currency** Enter the appropriate to-currency. This value is used with its associated From Currency field value as part of an exchange rate pair.

**Currency** This field displays when you are working with a rate definition with a rate category set to *Interest Rate*.

Select the currency for which you are creating an interest rate definition.

The From Currency and To Currency fields do not display.

**Day Count Basis** This field displays when you are working with an interest rate definition.

Select an interest basis:

*30/360*

*30E/360*

*Actual/360*

*Actual/365*

*Actual/Actual*

**Maximum Variance** Enter the percentage of variance that is allowed when a user maintains or overrides a market rate. If the change exceeds the tolerance, an error results. The default value is 2.50 (2.5%).

**Error Type** Select the type of error that results when the defined maximum variance is exceeded during data entry.

*None.* No error processing occurs and the new rate is used, even if it exceeds the maximum variance.

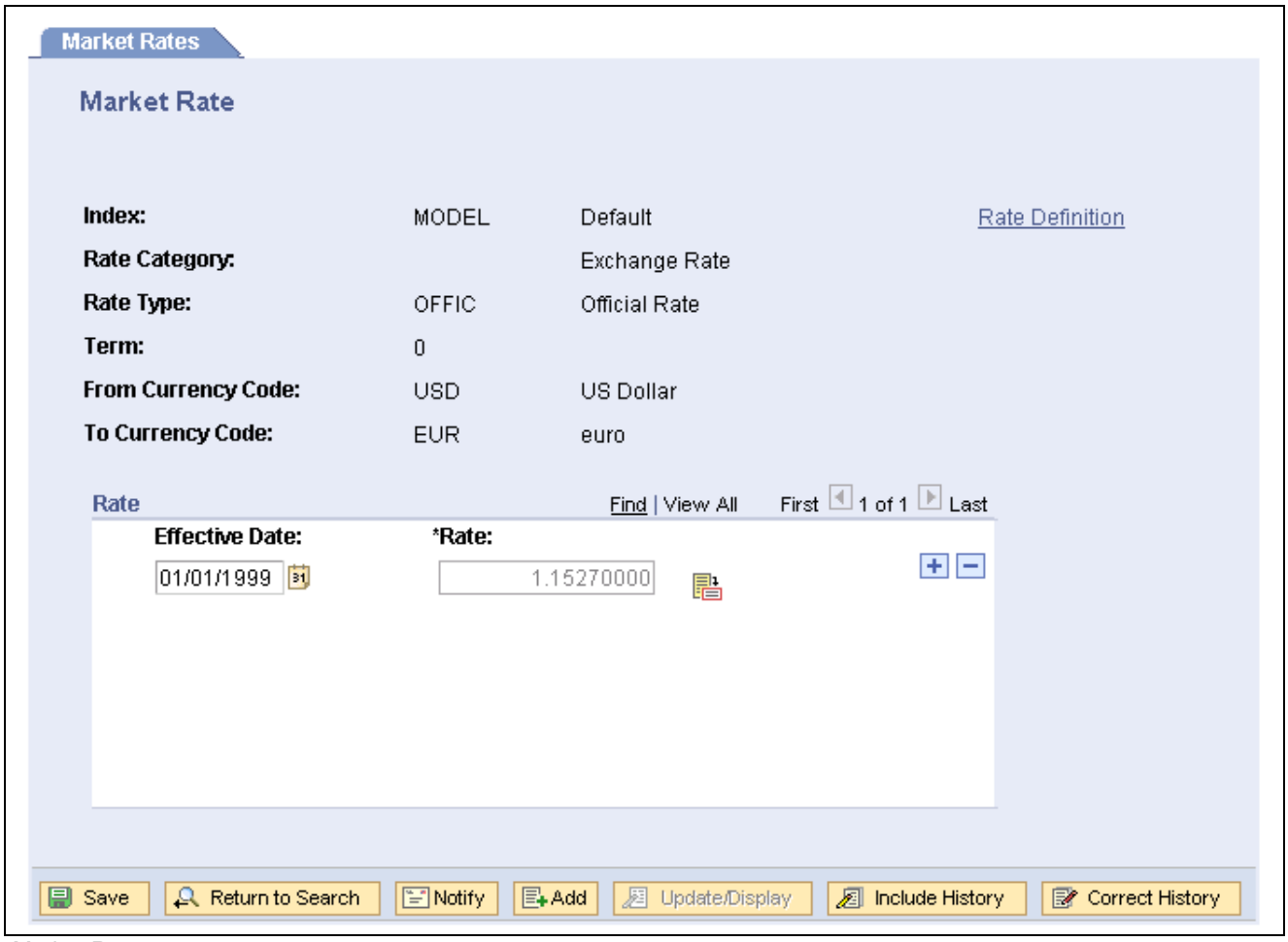
*Stop.* Processing halts and the system prevents you from saving the new rate.

*Warning.* This is the default value. A warning appears that you can ignore and proceed to save the new rate.

## Defining Market Rates

Access the Market Rates page.

Copyright © 1992-2006, Oracle. All rights reserved.




Market Rates page

The data you enter on this page is stored in the RT_RATE_TBL table, which is the common repository for all types of market rates including exchange rates and interest rates.

This page is not editable if all the following are true:

- The rate is triangulated.
- The primary visual rate is the cross rate.
- The Allow Override option is clear for the exchange rate's quotation method on the Currency Quotation Method page.

---

**Note.** When working with interest rates, the From Currency Code and To Currency Code fields may contain the same field value.

---

## Rate

**Rate**

Displays the visual rate. If you are working with a triangulated exchange rate, this field displays the primary visual rate, which is typically the cross rate, but can also be one of the other component rates of the triangle.

During online maintenance of market rates, you don't view or change RATE_MULT and RATE_DIV values directly, but instead access this visual rate, which is calculated by page logic based on RATE_MULT, RATE_DIV,


Copyright © 1992-2006, Oracle. All rights reserved.

and the currency quotation method defined for the currency pair on the Currency Quotation Method page. The visual rate is stored temporarily on a page work record.

**Exchange Rate Detail** Click to access the Exchange Rate Detail page, where you can view all three visual rates of a triangulated exchange rate.

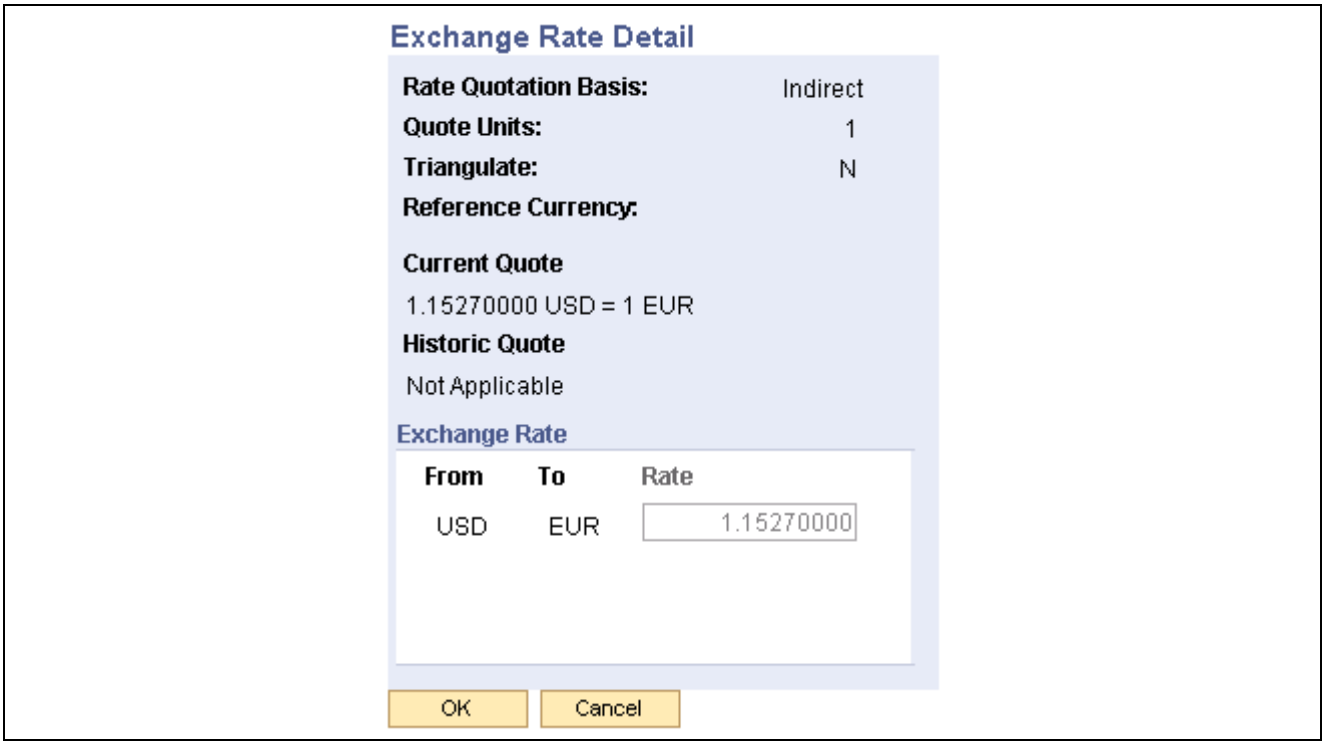
If a quotation method has been defined for the currency pair and the Auto Reciprocate option for the currency quotation method is selected, then creating or maintaining a rate for a currency pair on this page automatically creates or updates the rate for the reciprocal currency pair. For example, if you change the USD-to-GBP rate, the GBP-to-USD rate is automatically updated. You can only auto-reciprocate currency pairs for which currency quotation methods have been defined on the Currency Quotation Method page.

See Chapter 4, "Working With Currencies and Market Rates," Defining Currency Quotation Methods, page 48.

**Note.** The results of updating the rate definition do not take effect until you save, close, and reopen the Market Rates page.

## Accessing Exchange Rate Details

Access the Exchange Rate Detail page.




Exchange Rate Detail page

The primary record for this page is the Exchange Rate work record. For triangulated rates, you can update rate values for all three components of the triangulated rate.

**Rate Quotation Basis** Displays the quotation basis for the exchange rate as defined in the Currency Quotation Method page.

**Quote Units** Displays the quote units for the exchange rate as defined in the Currency Quotation Method page.

Copyright © 1992-2006, Oracle. All rights reserved.

**Triangulate** Displays the triangulation setting for the exchange rate as defined in the Currency Quotation Method page.

**Reference Currency** For triangulated exchange rates, displays the reference currency used in the triangulated exchange.

**Current Quote** Displays the current exchange rate used to convert the from-currency to the to-currency.

For a direct, non-triangulated rate, this field displays quote units (or 1) to the left side of the equal sign and the visual rate on the right. For example:

1 USD = 1.40000000 CAD

For an indirect, non-triangulated rate, this field displays the visual rate to the left of the equal sign and quote units (or 1) on the right. For example:

1.400000000 CAD = 1 USD

For a triangulated rate, this field displays the two component rates of the triangle: the rate for converting the from-currency to the reference currency (USD to EUR) and the rate for converting the reference currency to the to-currency (FRF to EUR). For example:

1.25 USD = 1 EUR = 6.8 FRF

**Historic Quote** If page logic determines that the exchange rate, as stored in the database, is inconsistent with the current quotation method, this field displays a quote based on the current quotation method, instead of the quotation method active on the rate effective date.

Data provided in the historic quote field allows you to see how the exchange rate has changed over time, using a consistent quotation method, even if the quotation method has actually changed.

For example, if you are viewing a historical rate where FRF was converted to USD directly using a calculated reciprocal rate of 1 FRF = 0.1470588 USD and the current quotation method for this currency pair is indirect, the conversion function recalculates the visual rate based on indirect quotation, that is 6.8000001 FRF = 1 USD.

This field also displays a quote if the historic quote method was non-triangulated and the current quote method is triangulated.

A historic quote is also displayed if you override a cross rate and bypass triangulation, because the exchange rate being used is inconsistent with the current quotation method.

If the system determines that the exchange rate is consistent with the current quotation method, the field displays *Not Applicable.*

**Exchange Rate** Displays a single visual rate for non-triangulated exchange rates, or all three component visual rates for triangulated exchange rates. The cross rate for triangulated exchange rates is editable only if the Allow Override option box is selected for the exchange rate on the Currency Quotation Method page.


Copyright © 1992-2006, Oracle. All rights reserved.

# Calculating Cross, Triangulated, and Reciprocal Rates

In this section, we discuss how to run the EOP_RATECALC Application Engine process to calculate cross, triangulated, and reciprocal rates.

## Understanding the EOP_RATECALC process

Run the EOP_RATECALC process to calculate rates and update the market rates table.

The process performs three functions:

- Generates cross rates for non-triangulated currency pairs.

  For example, an organization subscribes to a rate service that provides all rates respective to USD. Starting with a USD to Canadian dollar rate and a USD to Mexican peso rate, the system can calculate a new Canadian dollar to Mexican peso cross rate.

- Generates triangulated rates for triangulated currency pairs.

  For example, the EUR to an EPC (euro participating currency) fixed rate has been established on the market rate table and a new EUR to USD rate has just been entered. Using this information, the process can create a new USD to EPC triangulated rate. The difference between triangulated rates and cross rates affects how the data is stored in the database. When calculating a cross rate, you actually create a new rate. When calculating a triangulated rate, the individual components of the source rates are stored on the target.

- Generates reciprocal rates for those currency pairs that are not automatically reciprocated.

  For example, using a USD to CAD rate as the source, the process calculates the CAD to USD reciprocal. If currency quote methods are in place, the visual rate remains the same and there is a difference in how the data is stored in the database (RATE_MULT and RATE_DIV are inverse). If currency quote methods are not used, the process actually calculates an inverse rate, meaning that the visual rates will differ.

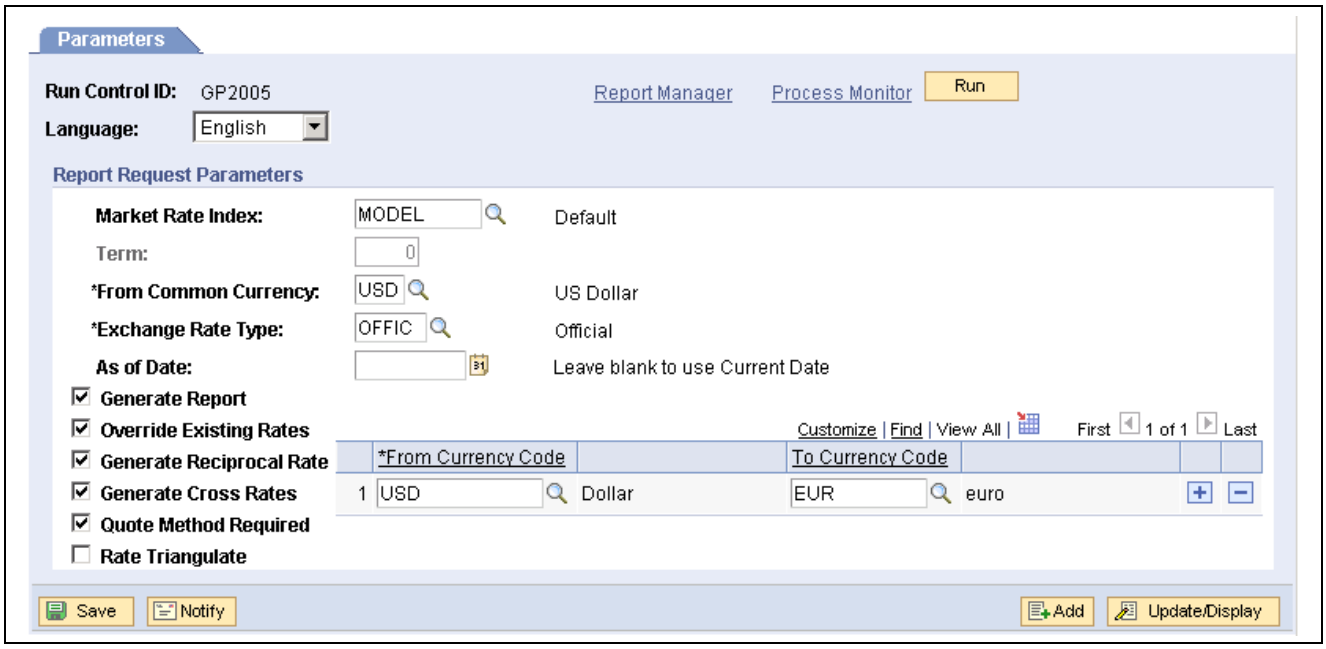
## Page Used to Calculate Cross, Triangulated, and Reciprocal Rates

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Cross/Reciprocal Rate Calc - Parameters (cross/reciprocal rate calculation – parameters) | EO_RATECALC | Depends on application. | Set run control parameters to run the EOP_RATECALC Application Engine process, which sets up cross, triangulated, and reciprocal rates. |

## Running the EOP_RATECALC Process

Access the Cross/Reciprocal Rate Calc - Parameters page.

Copyright © 1992-2006, Oracle. All rights reserved.




Cross/Reciprocal Rate Calc - Parameters page

**Market Rate Index** Select a market rate index. Applications other than PeopleSoft Treasury should use the default index that you select for the exchange rate.

**Term** This value defaults from the value entered on the Market Rate Definition page.

**From Common Currency** Select a currency code to calculate a reciprocal rate.

**Exchange Rate Type** Select the exchange rate type to use for this calculation

**As of Date** Select the effective date of the newly created exchange rates, which are the output of the process. The as of date also determines the rates used as the basis for the calculations, which are the input of the process.

The report uses the most current currency quotation method for the currency pair as the input to the process. If the as of date is the current effective rate on the specified date, it can affect triangulation. For example, a USD to EPC (euro participating currency) triangulated rate effective April 1, 2004 might be comprised of the EUR to USD rate also effective April 1, 2004 and the fixed EUR to an EPC rate effective on the date the newly participating EPC officially becomes a euro participating currency.

**Generate Report** Select to generate a report that displays the cross, triangulated, and reciprocal rate calculations performed by the process.

**Override Existing Rates** Select to have the calculated rates override rates for the exchange rate type, regardless of the as of date.

**Generate Reciprocal Rate** Select to calculates reciprocal rates for currency pairs that do not have the Auto Reciprocate option select on the Currency Quotation Method page.

You can select this option alone, or in combination with the Generate Cross Rates and Rate Triangulate options.

This process does not directly manipulate the exchange rates. The system uses numerator and denominator values instead, such that the following is true:


Copyright © 1992-2006, Oracle. All rights reserved.

(from-currency / RATE_DIV) × RATE_MULT = to-currency

Fro example, suppose you want a reciprocal rate between USD and CHF and assume a two-to-one ratio. If the exchange rate for USD to CHF is quoted directly (either using a direct quote method that you selected or using the system default), this rate is stored as RATE_MULT = 2 and RATE_DIV = 1. The rate is represented as 1 USD = 2 CHF, with a visual rate of 2.

In turn, the CHF to USD rate must be indirect. The reciprocal is a simple exchange, storing the rate as RATE_MULT = 1 and RATE_DIV = 2. The visual rate remains 2.

If quote methods are not being used, the CHF to USD rate must be quoted directly (the default), so the reciprocal rate is actually a calculated inverse. This rate is stored as RATE_MULT = 0.5 and RATE_DIV = 1, with a visual rate of 0.5.

In this example between USD and CHF, using a quote method and using a calculated inverse produced the same end result, 1/2 equals 0.5. But in actual practice, the manipulation of exchange rates is a major task and is one of the reasons for establishing the currency quote method.

**Generate Cross Rates**

Select to automatically generate cross rates. For example, to generate cross currency rates for USD, CAD, and MXP, you enter USD to CAD = 1.473 and USD to MXP = 9.8793. The system automatically generates CAD to MXP = 9.8793/1.473 = 6.7069246.

If you choose to generate cross rates, the From Cur (from-currency) and To Cur (to-currency) fields display and you must select a from-currency and a to-currency. You can enter a wild card of *%* in either or both fields to indicate from all or to all currencies.

**Rate Triangulate**

Select to convert two currencies through a third currency.

Select to convert two currencies through a third currency. If you select Rate Triangulate, the From Cur (from-currency) and To Cur (to-currency) fields display and you must select a from-currency and a to-currency. You can enter a wild card of *%* in either or both fields to indicate from all or to all currencies.

**Quote Method Required**

Select to indicate that you want the process to perform selected calculations only if the currency pairs have an existing currency quotation method definition.

# Using the Currency Exchange Calculator

In this section, we discuss how to convert amounts using the Currency Exchange Calculator.

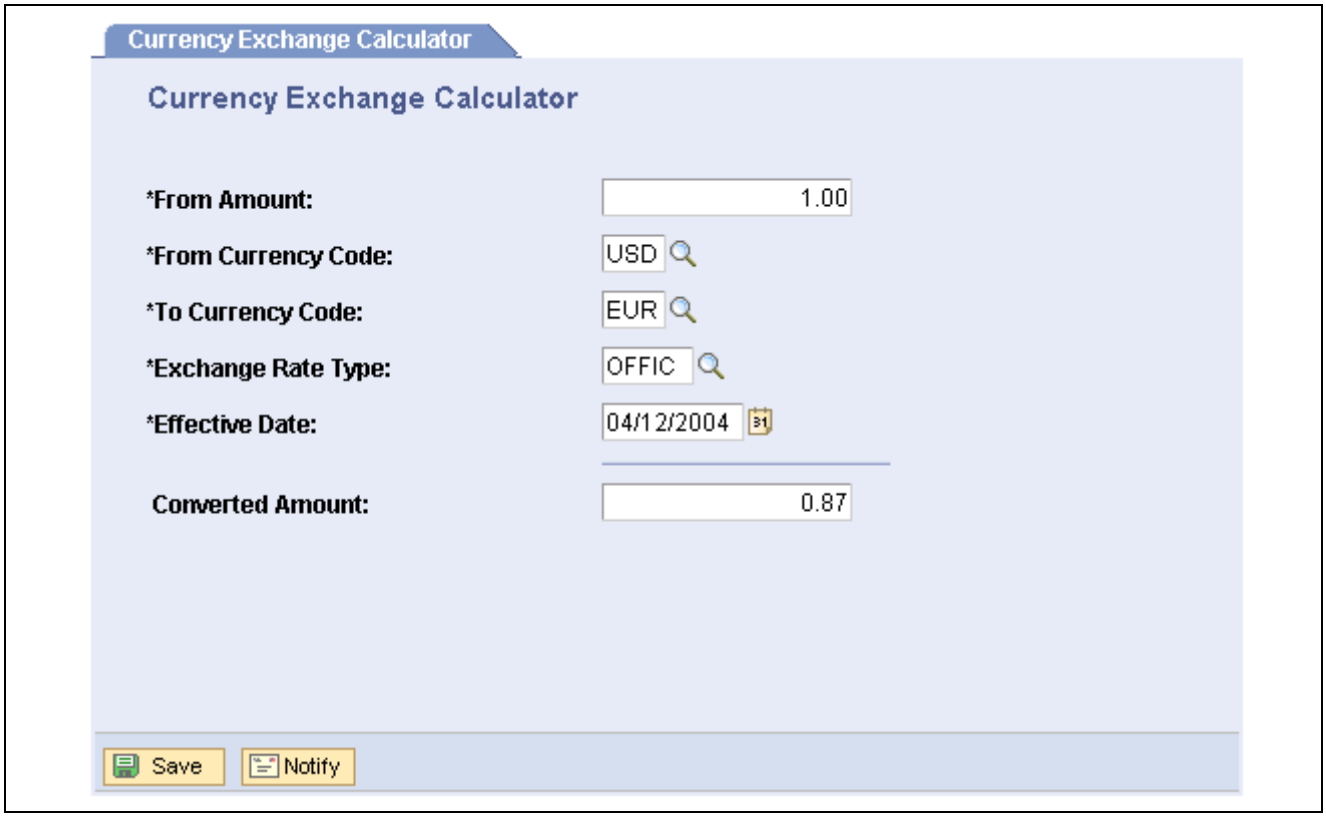
Copyright © 1992-2006, Oracle. All rights reserved.

# Page Used to Use the Currency Exchange Calculator

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Currency Exchange Calculator | CURRENCY_EXCHNG_PN | Depends on application. | Calculate currency exchange between currencies. This tool enables you to select a rate type other than the base currency, but does not enable you to override the exchange rate. |

# Converting Amounts Using the Currency Exchange Calculator

Access the Currency Exchange Calculator page.




Currency Exchange Calculator page

**From Amount** The currency exchange is based on the from amount that you enter and the current exchange rate set up on the Market Rates page.

See Chapter 4, "Working With Currencies and Market Rates," Defining Market Rates, page 55.

**From Currency Code** Select the currency code from which to calculate the exchange amount.

**To Currency Code** Select the currency code to which to calculate the exchange amount.

**Exchange Rate Type** Select the type of exchange rate to use for this calculation.

**Converted Amount** Click Save to calculate the amount and display it in this field.


Copyright © 1992-2006, Oracle. All rights reserved.

# CHAPTER 5

# Using Datasets

This chapter provides an overview of datasets and discusses how to:

- Define dataset rules.
- Define dataset roles.
- Define mobile data distribution.

## Understanding Datasets

Datasets enable role-based filtering and distribution of data. You can limit the range and quantity of data displayed for a user by associating dataset rules with a user's dataset roles. The result of dataset rules is a set of data appropriate to the user's dataset roles.

You can also limit the range and quantity of data passed to a mobile device by defining data distribution rules based on datasets. Data distribution rules define the selection of data downloaded to a mobile device. The dataset may differ depending on the mobile device.

**Note.** If you are using PeopleCode to control data distribution, consider using datasets instead.

## Defining Dataset Rules

This section provides an overview of dataset rules and discusses how to create dataset rules.

### Understanding Dataset Rules

Dataset rules define datasets for use in conjunction with each dataset role's security rules. Defining dataset rules creates Structured Query Language (SQL) statements that select the dataset displayed for each rule.

To use dataset rules and roles:

1. Define dataset rules, which are based on a synchronized record.

   You define a dataset rule to return a subset of rows from the selected synchronized record based on the dataset role to which you will link the rule.

   These dataset rules are based on views that can join to any record in your PeopleSoft system.

   For each rule condition, the user specifies a field that comes from the search record name defined in the dataset rule. When the specified field has neither a prompt or translate table edit, the following system variables, which are delivered as system data to all applications, can be used to filter the condition:

   - %Blank

Copyright © 1992-2006, Oracle. All rights reserved.

- %Date
- %EmployeeID
- %PersonID
- %Time
- %UserID

2. Assign the dataset rules to dataset roles according to role security and data requirements.

   Each dataset role can have multiple dataset rules. You can use existing dataset roles, or create new dataset roles by selecting from existing PeopleTools security-based user roles.

   See *Enterprise PeopleTools 8.48 PeopleBook: Security Administration,* "Setting Up Roles"

3. Ensure that the original user roles on which dataset roles were based are associated with appropriate user IDs.

   Each user ID can have multiple user roles.

   See *Enterprise PeopleTools 8.48 PeopleBook: Security Administration,* "Administering User Profiles," Setting Roles

## Page Used to Define Dataset Rules

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Dataset Rules | EOEC_DATASET | Enterprise Components, Component Configurations, Datasets, Dataset Rules | Define the rules that make up a dataset. |

## Creating Dataset Rules

Access the Dataset Rules page.

The number of rule conditions in a dataset rule is limited only by your performance requirements. You can set a series of rule conditions that can navigate through as many records as necessary.

### Dataset Rules

**Search Record Name** Select the name of the search record for this rule. You can create a view specifically for use in the rule.

**Status** Select *Active* or *Inactive.*

### Rule Conditions

**…((** and **))…** If the AND or OR field is left blank, specify the nesting level for this condition. Be sure to match opening and closing parentheses.

**Field Name** Select the field name on which this rule operates.

**Operator** Specify the operation with which to compare the specified field value. Select from standard conditional operators.

**Field Value** Specify the value of the specified field against which to compare.

 Copyright © 1992-2006, Oracle. All rights reserved.

**AND or OR** For second and subsequent rule conditions, specify *AND* or *OR,* or leave blank if the rule statements are nested.

**Test SQL** Click to test the validity of the rule conditions. The result is returned below the button.

**Show SQL** Click to view the SQL statement generated by the rule.

# Defining Dataset Roles

Set up user roles by associating dataset rules with user roles.

## Page Used to Define Dataset Roles

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Dataset Roles | EOEC_MP_ROLE | Enterprise Components, Component Configurations, Datasets, Dataset Roles | Define dataset roles that associate existing PeopleTools user roles with dataset rules. |

## Defining Dataset Roles

Access the Dataset Roles page.

Select an existing dataset role for editing, or create a new dataset role by selecting from existing PeopleTools security-based user roles.

See *Enterprise PeopleTools 8.48 PeopleBook: Security Administration,* "Setting Up Roles"

**Dataset Name** Select the dataset with which the component rule is associated.

**Rule** Select the component rule.

**Laptop** and **PDA** Select to display the resulting data on a laptop computer or PDA.

**Note.** If neither Laptop nor PDA is selected, no data from this rule is displayed.

# Defining Mobile Data Distribution

Use datasets to define the data distributed to mobile devices running PeopleTools Mobile Agent. This section provides an overview of mobile data distribution and discusses how to:

- Define mobile data distribution rules.
- Use mobile user rules.

Copyright © 1992-2006, Oracle. All rights reserved.

# Understanding Mobile Data Distribution

Mobile devices can have limited processing power, storage capacity, and display space. You can limit the range and quantity of data passed to the mobile device by associating dataset rules with synchronizable component interfaces. Mobile data distribution rules define the selection of data from network servers for download to a mobile device. The result of data distribution rules is a set of data appropriate to the user's roles. The set of data may differ depending on the mobile device.

Data distribution for mobile applications implements security and filters the data downloaded to the mobile device. You define data distribution for mobile devices based on datasets by selecting dataset rules assigned to the mobile device user's dataset roles.

## Implementing Mobile Data Distribution

To filter data defined by dataset rules to mobile devices, developers must include the function DistributeDataByRules() in the synchronizable component interface's OnSelect PeopleCode method.

For example:

```
Declare Function DistributeDataByRules PeopleCode
FUNCLIB_ECMOBIL.EOEC_ONSELECT FieldFormula;
DistributeDataByRules();
```

## See Also

*Enterprise PeopleTools 8.48 PeopleBook: PeopleSoft Mobile Agent*

*Enterprise PeopleTools 8.48 PeopleBook: PeopleCode Language Reference*

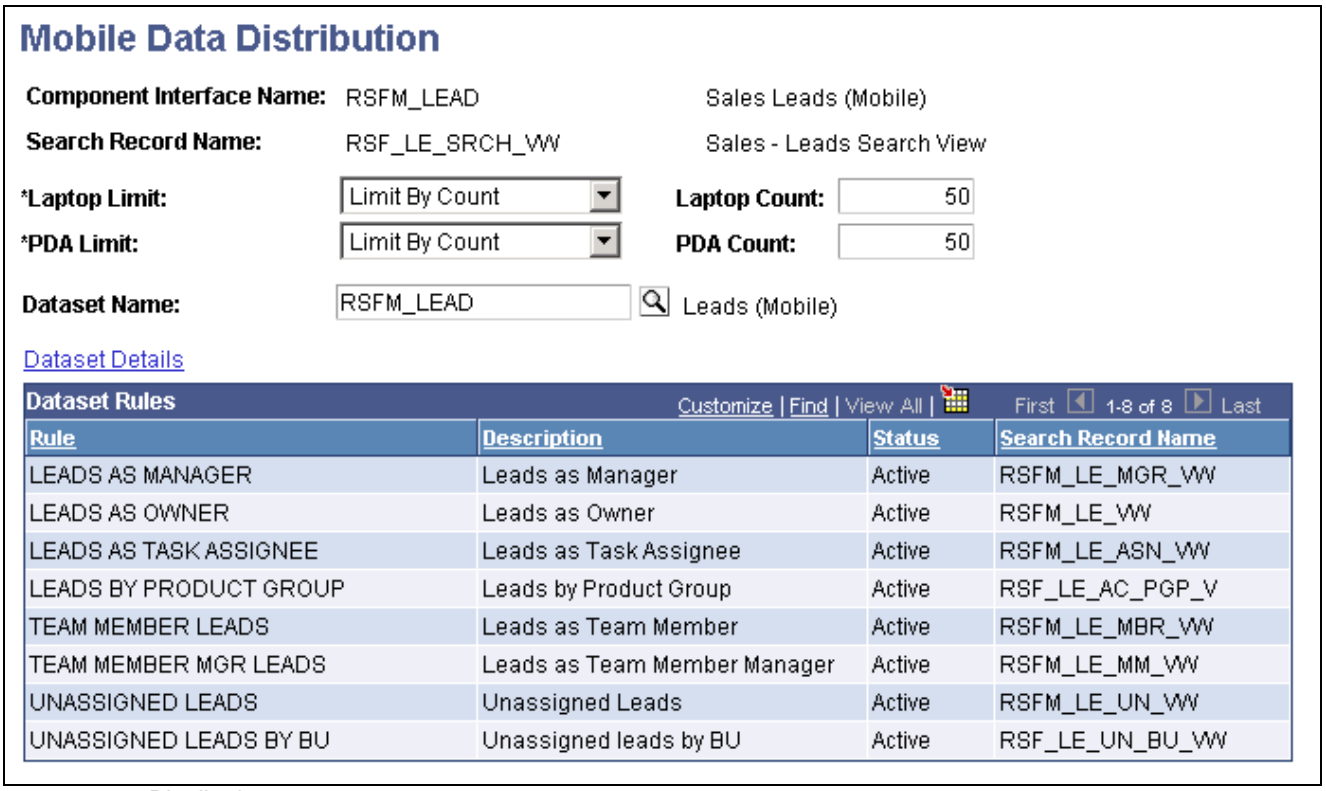
# Pages Used to Define Mobile Data Distribution

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Mobile Data Distribution | EOEC_MP_RULE | Enterprise Components, Component Configurations, Mobile, Mobile Data Distribution | Define data distribution rules for mobile devices based on datasets. |
| Mobile User Rules | EOEC_MP_USRRULE | Enterprise Components, Component Configurations, Mobile, Mobile User Rules | Preview the effect of mobile data distribution. |

# Defining Mobile Data Distribution Rules

Access the Mobile Data Distribution page.

 Copyright © 1992-2006, Oracle. All rights reserved.




Mobile Data Distribution page

**Note.** Component interfaces selected for mobile data distribution must be synchronizable; only synchronizable component interfaces are available from the prompt.

**Laptop Limit**

Select the limiting factor for data instances to be downloaded to a laptop computer during synchronization. Select from:

*All Data:* Select to download all data matching the rule's conditions.

*Limit By Count:* Select to download only data matching the rule's conditions up to the specified count. The count is based on the number of rows returned. Referenced data is also included.

*Referenced Data Only:* Select to download only data only if it is referenced by another component. You would be more likely to select this option for setup data than for transaction data.

For example, if you have 50,000 products, but the leads you download are related to only 2,000 of the products, select *Referenced Data Only* so that you only download the 2,000 products, instead of the entire set.

If *Reference Data Only* is selected for products, downloads do not include products unless other objects such as leads and opportunities that reference products are downloaded.

**PDA Limit**

Select the limiting factor for data instances to be downloaded to a PDA during synchronization. Select from:

*All Data:* All data matching the rule's conditions is downloaded.

*Limit By Count:* Only data matching the rule's conditions up to the specified count is downloaded. Referenced data is also included.

Copyright © 1992-2006, Oracle. All rights reserved.

*Referenced Data Only:* Only data referenced by the component interface is downloaded.

**Laptop Count** and **PDA Count** — If you select *Limit By Count* in the Laptop Limit field or the PDA Limit field, specify the maximum number of data instances to be downloaded.

**Dataset Name** — Select the dataset to apply to this mobile data distribution rule.

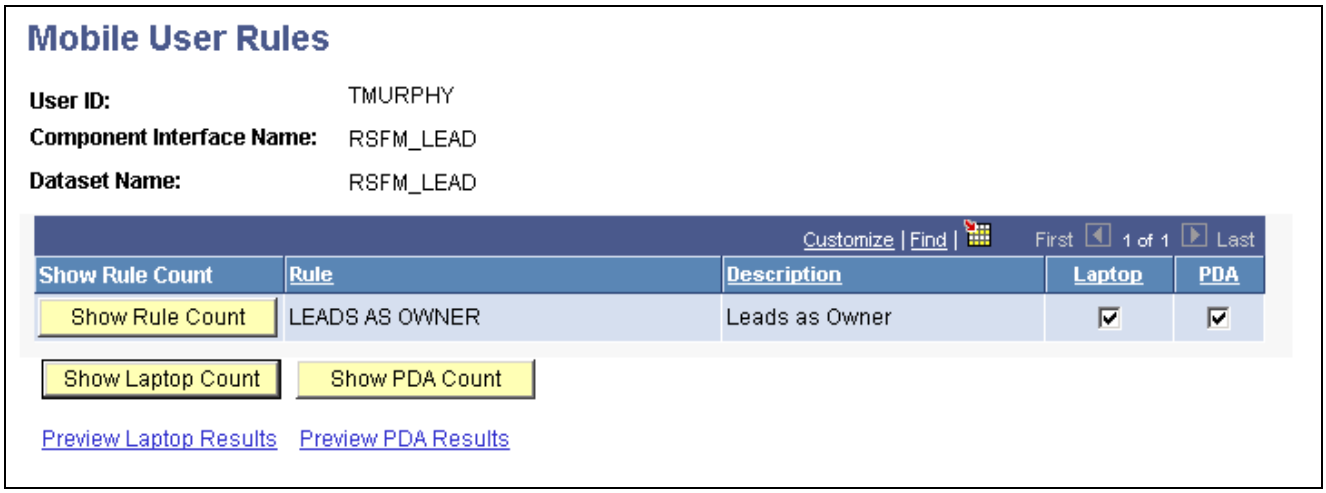
**Dataset Details** — Click to access the Dataset Rules page, where you can view and modify the selected dataset definition.

See Chapter 5, "Using Datasets," Creating Dataset Rules, page 64.

The Dataset Rules grid lists rules for the specified dataset.

# Using Mobile User Rules

Access the Mobile User Rules page.




Mobile User Rules page

A mobile user can specify whether a selected data distribution rule returns data to a selected mobile device. The user must be signed in with a user ID, not as an administrator, to define mobile user rules.

**Show Rule Count** — Click to view the number of results the mobile data distribution rule returns.

**Show Laptop Count** and **Show PDA Count** — Click to view the number of results the mobile data distribution rule returns to a laptop computer or PDA based on any limits set for the mobile device on the Mobile Data Distribution page.

See Chapter 5, "Using Datasets," Defining Mobile Data Distribution, page 65.

**Laptop** and **PDA** — Select to display the results of this mobile data distribution rule on a laptop computer or on a PDA, or both.

**Note.** If neither Laptop nor PDA is selected, no data from this mobile data distribution rule is displayed.

**Preview Laptop Results** and **Preview PDA Results** — Click to preview the data that will be downloaded to a laptop or PDA as a result of this mobile data distribution rule and any limits set for the mobile device on the Mobile Data Distribution page.


Copyright © 1992-2006, Oracle. All rights reserved.

# CHAPTER 6

# Using Interactive Reports

This chapter provides an overview of PeopleSoft Business Analysis Modeler Interactive Reports and discusses configuration of the Interactive Report environment.

---

**Note.** Each PeopleSoft application that delivers an Interactive Report provides documentation for viewing that specific report.

---

## Understanding Interactive Reports

PeopleSoft Business Analysis Modeler is a calculation and analysis tool used by various PeopleSoft applications. Interactive Reports use PeopleSoft Business Analysis Modeler to calculate and present multidimensional data to the user.

Interactive Reports include the following:

- Pages to define and configure Interactive Reports.
- PeopleSoft queries to select the appropriate data for the report.
- The PeopleSoft Business Analysis Modeler model file containing the data structure and calculation rules.
- The PeopleSoft Business Analysis Modeler PageView file containing the report layout.
- A page to capture the capture query prompt values and launch the report.

## Common Elements Used in This Chapter

| | |
|---|---|
| **Report ID** | Unique Interactive Report identifier assigned by the report developer. |
| **Description** | Describes the associated report, server, and database connection, depending on which page this field appears. |
| **Last Update Date/Time** | Lists the user and date the page was most recently updated. |

Copyright © 1992-2006, Oracle. All rights reserved.

# Configuring Interactive Reports

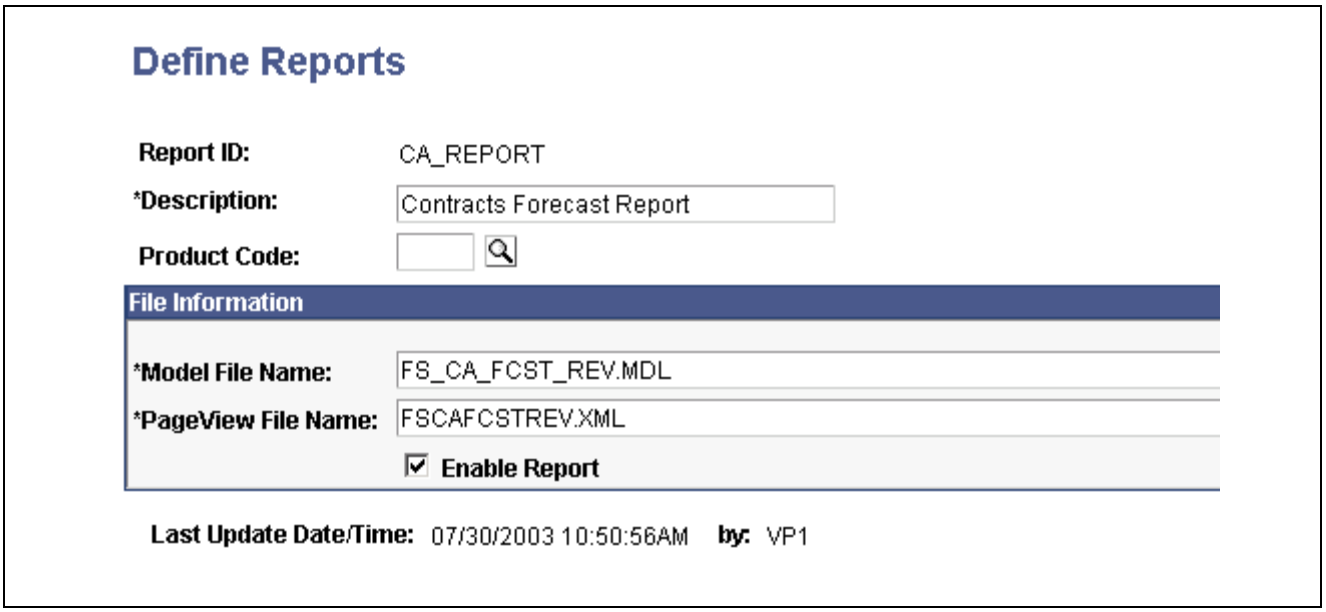
## Pages Used to Configure Interactive Reports

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Define Reports | EOBF_REPORT | Enterprise Components, Interactive Report Definitions, Reports | Defines the report file names. |
| Define Servers | EOBF_SERVER | Enterprise Components, Interactive Report Definitions, Servers | Defines web server and file paths. |
| Define ODBC Connections | EOBF_ODBC | Enterprise Components, Interactive Report Definitions, ODBC Connections | Defines the ODBC connection string to the PeopleSoft database. |
| Environment Configuration | EOBF_CFG_REPORT | Enterprise Components, Interactive Report Definitions, Environment Configuration | Defines the PeopleSoft Business Analysis Modeler environment for a specific report. |
| Map Queries | EOBF_QRY_NAME | Enterprise Components, Interactive Reports Definition, Queries | Maps specific PeopleSoft queries to import maps in the PeopleSoft Business Analysis Modeler model. Import map entries on this page have been defined by the PeopleSoft application delivering the report. |
| Base Language Query Prompt | EOBF_QRY_PRMPT | Click the View Query Prompts link for the Query Name (Base Language) field on the Map Queries page. | Specifies the record and field containing the base language query prompt values when the report is launched. |
| Related Language Query Prompt | EOBF_QRY_PRMPT_LNG | Click the View Query Prompts link for the Query Name (Related Language) field on the Map Queries page. | Specifies the record and field containing the related language query prompt values when the report is launched. |
| Validate Environment | EOBF_ENV_UTILITY | Enterprise Components, Interactive Reports Definition, Validate Environment | Validates the connection of the architectural components at installation. |

## Defining Reports

Access theDefine Reports page.


Copyright © 1992-2006, Oracle. All rights reserved.

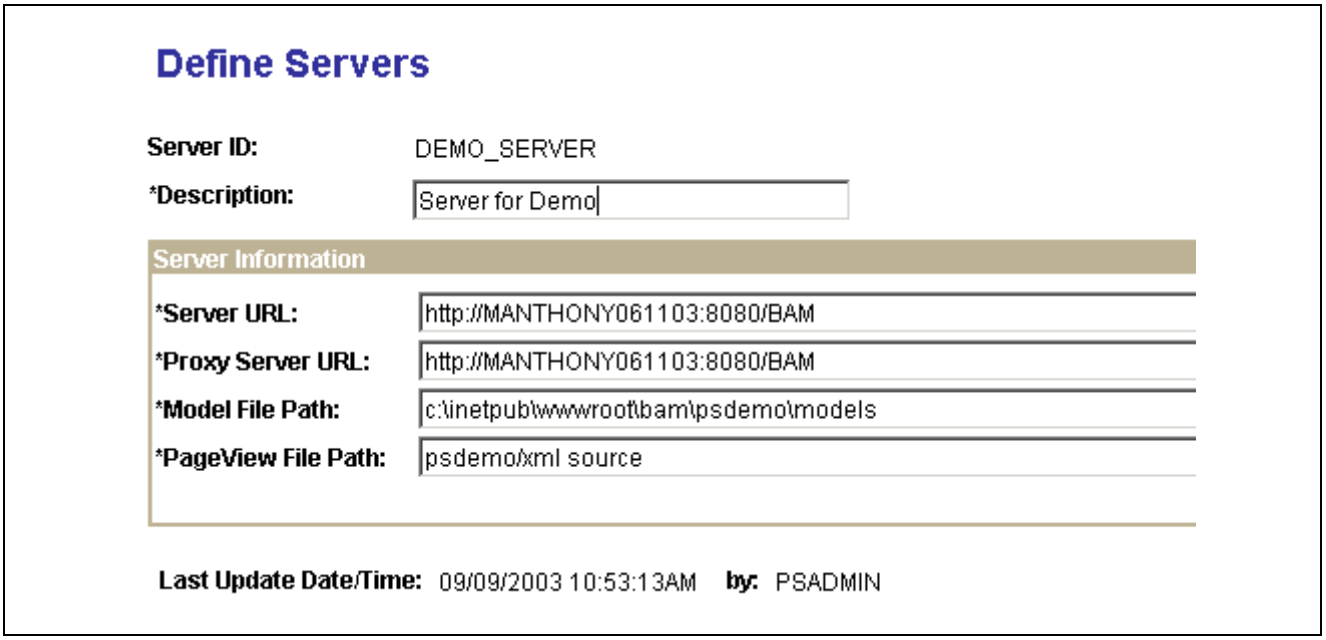



Define Interactive Reports

| | |
|---|---|
| **Product Code** | (Optional) Code of the PeopleSoft licensed product associated with the report. |
| **Model File Name** | File name of the PeopleSoft Business Analysis Modeler model file delivered for the report. |
| **Page View Name** | File name of the PeopleSoft Business Analysis Modeler PageView file delivered for the report. |
| **Enable Report** | Select this checkbox to allow valid users access to the report. |

## Defining Servers

Access the Define Servers page.




Define Servers

Copyright © 1992-2006, Oracle. All rights reserved.

**Server ID** Unique identifier of the PeopleSoft Business Analysis Modeler Analytic Web Server used for Interactive Reports.

**Server URL** and **Proxy Server URL** Location of the PeopleSoft Business Analysis Modeler Analytic Web Server and proxy server. Include the port specification if its value is other than 80. If there is no proxy server, enter the Server URLProxy Server URL field.

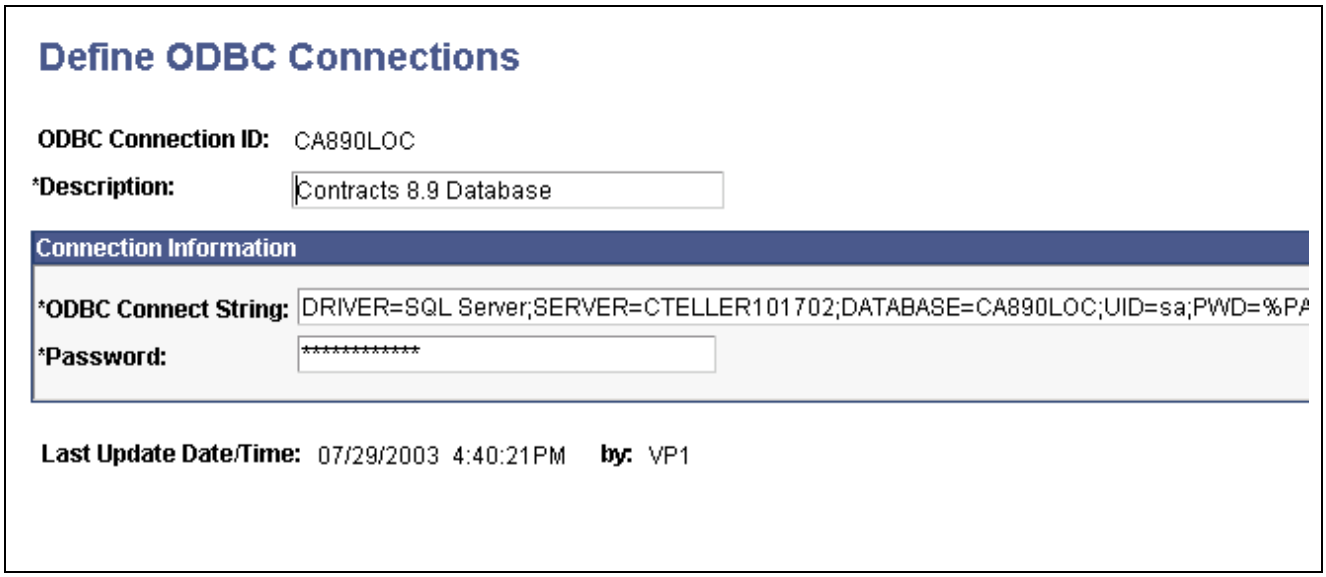
**Model File Path** Path of the .MDL (model) file shown on the Define Reports page. These files are usually on the PeopleSoft Business Analysis Modeler Analytic Web Server in the c:\inetpub\wwwroot\bam*PeopleSoft product*\models directory.

**PageView File Path** Path of the .XML file shown on the Define Reports page. These files are usually on the PeopleSoft Business Analysis Modeler Analytic Web Server in the c:\inetpub\wwwroot\bam*PeopleSoft product*\xml source directory.

**Note.** The PageView File Path is a relative path under c:\inetpub\wwwroot\BAM \ with forward slashes indicating subdirectories.

# Defining ODBC Connections

Access the Define ODBC Connections page.




Defining ODBC Connections

**ODBC Connection ID** Unique identifier of the ODBC Connect String.

**ODBC Connect String** Used for queries to the PeopleSoft database. The exact connect string syntax depends on the database platform and is listed in the following table.

**Password** and **Confirm Password** Enter a password to connect to the database listed in the ODBC Connect String. Reenter the password in the Confirm Password field. The user ID listed must have a minimum of *Select* privileges on the database.

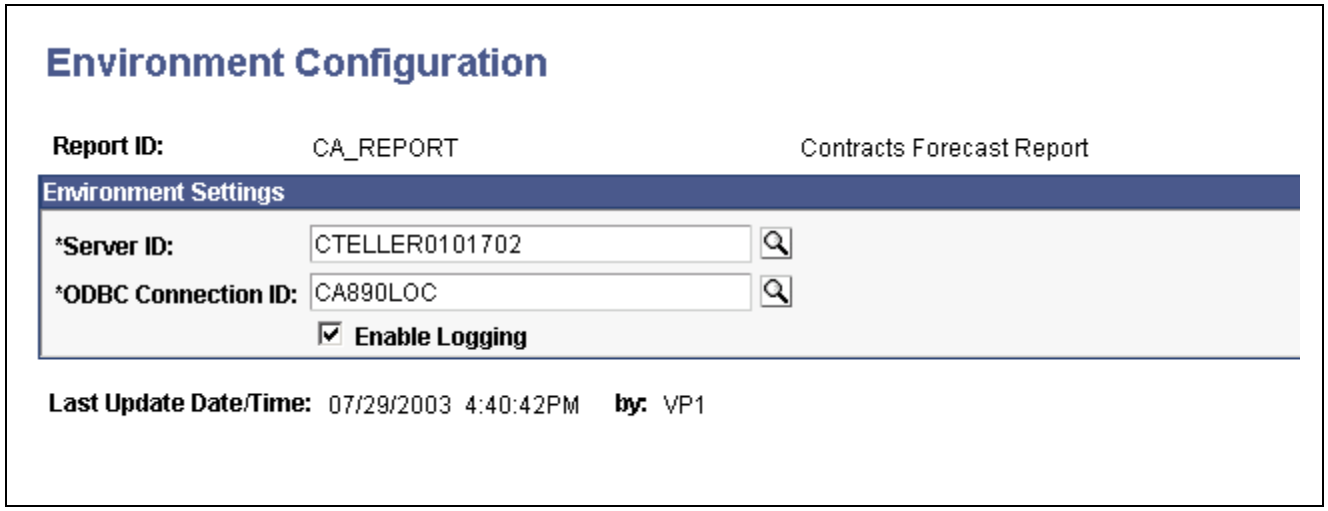
The following table lists the syntax to use for the various database platforms.

 Copyright © 1992-2006, Oracle. All rights reserved.

| Database Platform | Connection String Syntax |
|---|---|
| Microsoft SQL Server | DSN=*Data Source*;SERVER=*Server Name*;DATABASE=*Database Name*;UID=*UserID*;PWD=%PASSWORD%; |
| Oracle | DSN=*Data Source*;UID=*UserID*;PWD=%PASSWORD%; DBTYPE=ORACLE; |
| DB2 Unix | DSN=*Data Source*;UID=*UserID*;PWD=%PASSWORD%; SCHEMA=*Schema Owner*; |
| DB2 OS390 | DSN=*Data Source*;UID=*UserID*;PWD=%PASSWORD%;SCHEMA=*Schema Owner*;SQLID=*Database Name*; |
| Sybase | DSN=*Data Source*;UID=*UserID*;PWD=%PASSWORD%;DBTYPE=SYBASE; |
| Informix | DSN=*Data Source*;UID=*UserID*;PWD=%PASSWORD%; |

## Configuring the Environment

Access the Environment Configuration page.




Environment Configuration

Select the Server ID and ODBC Connection ID to be used with this report.

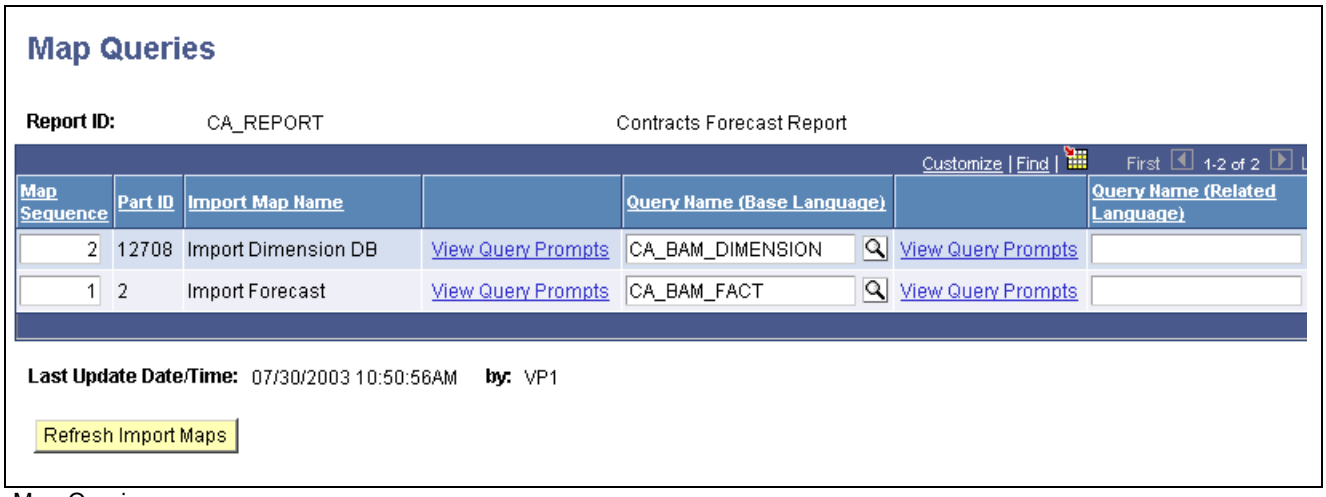
**Enable Logging** Select this checkbox to have all PeopleSoft Business Analysis Modeler Web Server commands logged. The default location for this log isc:\inetpub\wwwroot\bam\logyyyymmdd.txt. You can modify the location of the log during installation.

See *PeopleSoft Business Analysis Modeler Installation Documentation*

## Mapping Queries

Access the Map Queries page.

Copyright © 1992-2006, Oracle. All rights reserved.




Map Queries

**Map Sequence** Defines the order in which the queries are executed. Order can be important in the validation of data integrity.

**Part ID** Used to identify PeopleSoft Business Analysis Modeler Import Maps. This value is retrieved from the .MDL file.

**Import Map Name** Import Map name in the related .MDL file.

**View Query Prompts** If you select this link, a Query Prompt page appears listing prompts for the query.

**Query Name (Base Language)** Identifies the PeopleSoft query used for the selection of data to the Import Map.

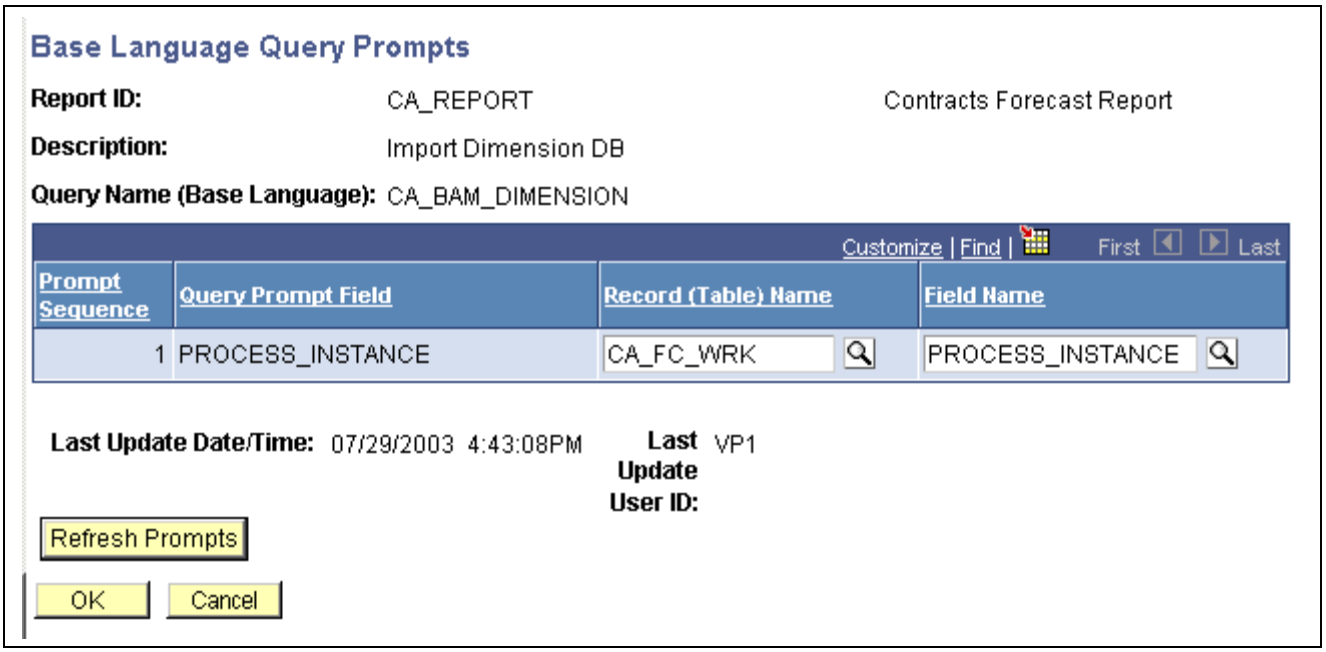
**Query Name (Related Language) (Optional)** If the report is used in a multilingual environment, identifies the PeopleSoft query which retrieves translated descriptions used in the Interactive Report.

**Refresh Import Maps** Clicking this retrieves a new list of import maps from the .MDL file and deletes current entries on the page.

# Viewing Query Prompts

Access the Base Language Query Prompt or Related Language Query Prompt secondary page.


Copyright © 1992-2006, Oracle. All rights reserved.

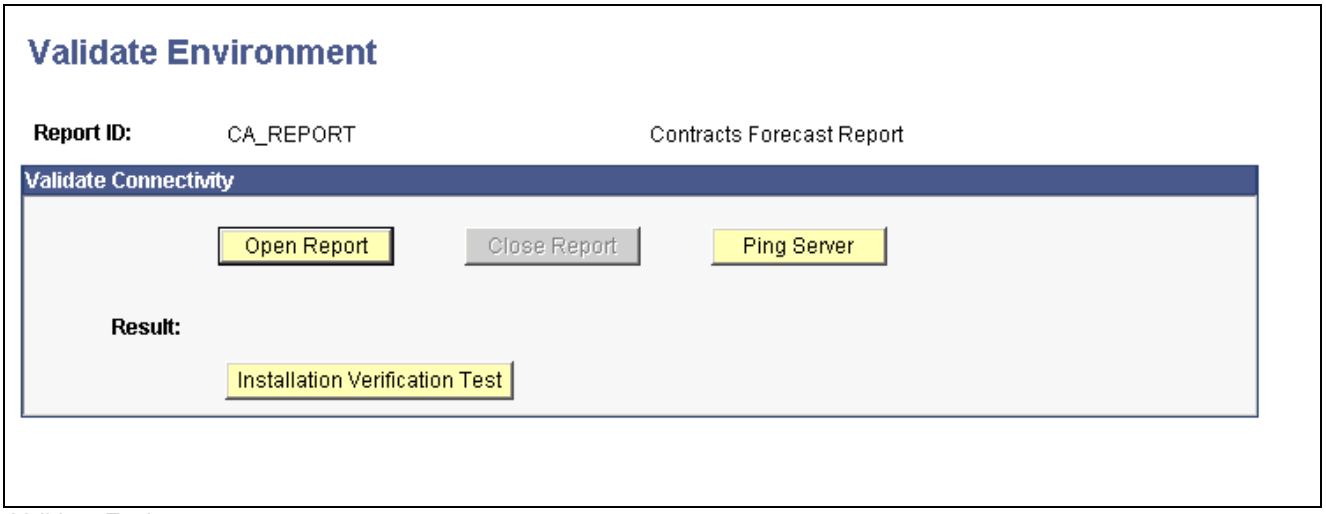



Base Language Query Prompt

| | |
|---|---|
| **Prompt Sequence** | Displays the order of the prompt in the query definition. |
| **Query Prompt** | Displays the prompt field in the query definition. |
| **Record (Table) Name** and **Field Name** | Specifies the record name and field name containing the prompt value. |
| **Refresh Prompts** | Clicking this retrieves a new list of prompts and deletes current entries in the page. |

# Validating the Environment

Access the Validate Environment page.




Validate Environment

Copyright © 1992-2006, Oracle. All rights reserved.

| | |
|---|---|
| **Open Report** and **Close Report** | Clicking Open Report opens the .MDL file and performs a checkout on that model. Clicking Close Report removes the checkout created. The Result field displays a message indicating success or failure for both operations. |
| **Ping Server** | Clicking this sends a command to ping the PeopleSoft Business Analysis Modeler Analytic Web Server. The Result field displays a message indicating success or failure. |


Copyright © 1992-2006, Oracle. All rights reserved.

# CHAPTER 7

# Setting Up the Credit Card Interface

This chapter provides an overview of credit card processing and discusses how to:

- Set up the credit card interface.
- Configure the Java interlink plug-in.
- Configure the PeopleSoft Application Designer plug-in.
- Set transaction inputs and outputs.

### See Also

*Enterprise PeopleTools 8.48 PeopleBook: PeopleSoft Business Interlinks,* "Business Interlinks for Runtime Plug-in Programming"

*Enterprise PeopleTools 8.48 PeopleBook: PeopleSoft Business Interlinks,* "Business Interlinks for Application Developers"

## Understanding Credit Card Processing

Use the credit card processing interface to integrate with credit card processing vendors. You have the following options for credit card processing:

- CyberSource integration.
- XML-based integration.
- Manual processing.

### Understanding a CyberSource Integration

If you use CyberSource, agents processing credit cards can use an application-specific credit card transaction page to submit the transaction to CyberSource for authorization, billing, authorization and billing, or credit. You can choose which types of transactions to permit.

Integration with other vendors may require customizations. There are third-party applications, such as TouchNet, that conform to CyberSource's API. TouchNet follows CyberSource's API standard.

The Credit Card Authorize Bill and Credit integration point enables interaction between PeopleSoft applications and CyberSource by way of the EOEC_CCI_TRANSACTION business interlink. Activate the business interlink by entering a CyberSource merchant ID on the Credit Card Interface Installation page.

PeopleSoft does not ship CyberSource code or programs. Contact CyberSource to contract for its services and obtain a CyberSource merchant ID and supporting files. When contacting CyberSource, identify yourself as a PeopleSoft customer.

Copyright © 1992-2006, Oracle. All rights reserved.

**Note.** PeopleSoft applications use PeopleSoft Business Interlink architecture to interface with CyberSource and CyberSource-compliant credit card processing vendors. To configure interfaces with unsupported vendors, create an interlink plug-in.

### See Also

http://www.cybersource.com

## Understanding an XML-Based Integration

PeopleSoft delivers the following XML messages for use with XML-based credit card processing vendors. Ensure that the XML message is transformed into the format that the vendor is expecting.

**Note.** If you have upgraded from a previous PeopleTools 8.48x release, the upgrade program creates service operations for these messages. The service operation names and message names are the same.

| Message Name | Description |
|---|---|
| EOEC_CCI_SYNC | Synchronous request that the credit card interface sends to the third-party vendor. The request can be for an authorize, bill, authorize and bill, or credit transaction. |
| EOEC_CCI_RESPONSE | The response to the request the credit card interface receives from the third-party vendor. |

Agents can then process credit cards using their application-specific credit card transaction page to submit the transaction to the vendor for authorization, billing, authorization and billing, or credit. You can choose which types of transactions to permit.

If you use an XML-based interface, you must use PeopleSoft Integration Broker to set up the node to which the XML messages should be sent and to indicate where the processor is located.

In configuring the node, specify that the node type is *External,.*. Also specify the authentication option that you arranged with the credit card processor.

### See Also

*Enterprise PeopleTools 8.48 PeopleBook: PeopleSoft Integration Broker*

*Enterprise PeopleTools 8.48 PeopleBook: Security Administration,* "Setting up Digital Certificates and Single Signon"

## Understanding Manual Processing

If you decide not to use CyberSource or an XML-based credit card processing system, then your application-specific credit card transaction page captures information for use with your own solution for processing credit card payments.

To support manual processing, you must build your own process to read credit card data from the PeopleSoft tables and process the credit card transaction.

 Copyright © 1992-2006, Oracle. All rights reserved.

# Setting Up the Credit Card Interface

This section discusses how to:

- Define connection parameters.
- Define accepted credit card types.
- Test the credit card interface.
- Test the credit card transaction.

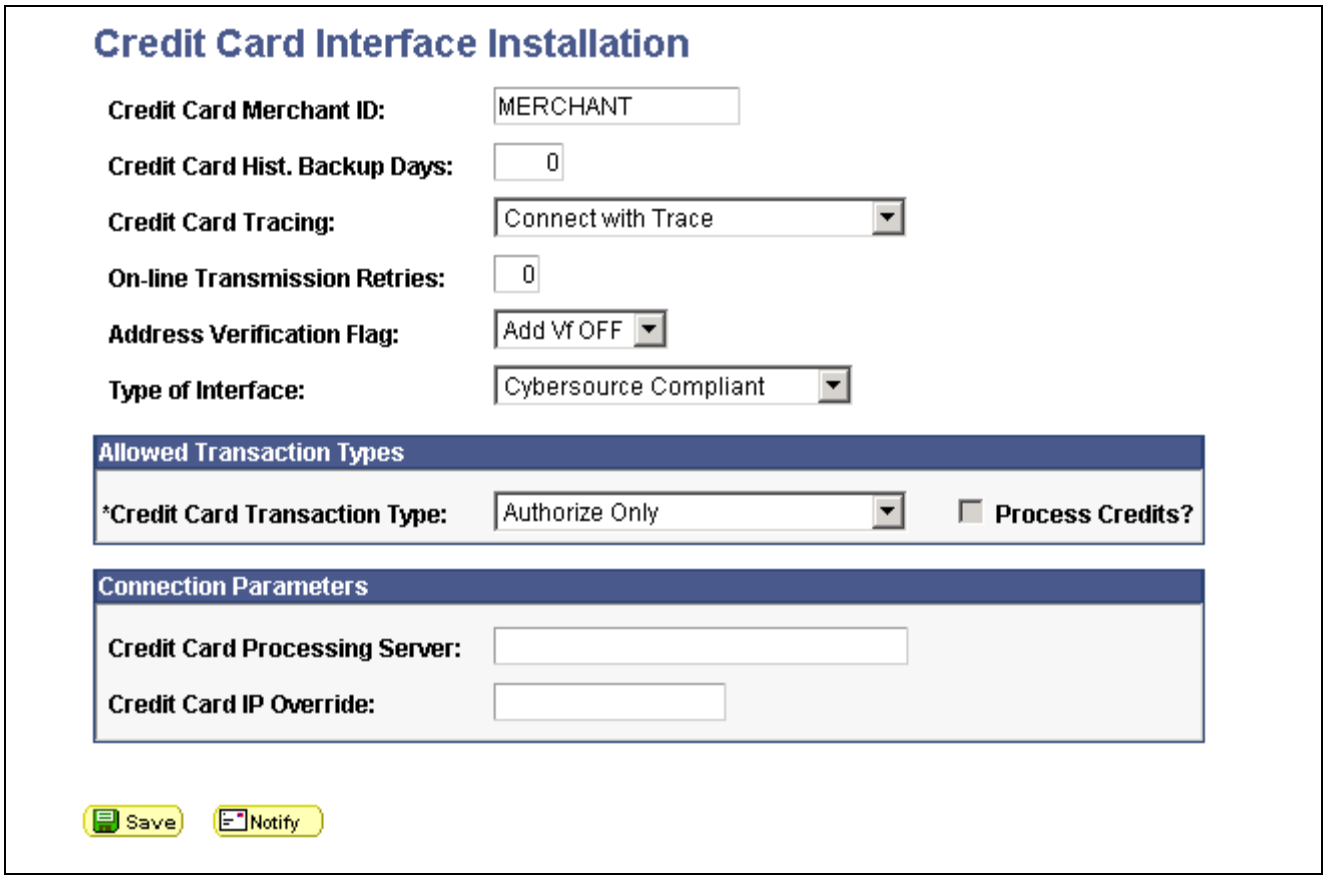
## Pages Used in Setting Up the Credit Card Interface

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Credit Card Interface Installation | EOEC_CCI_INSTAL | Enterprise Components, Component Configurations, Credit Card Interface, Setup | Set up connection parameters for credit card processing calls to CyberSource, or a CyberSource-compliant or XML-based third-party credit card processing vendor.<br>Before you set up credit card processing options, establish your merchant account with CyberSource, or a CyberSource-compliant or XML-based vendor. |
| Credit Card Setup | EOEC_CCI_CARDTYPE | Enterprise Components, Component Configurations, Credit Card Interface, Credit Card Types | Define the types of credit cards you accept for credit card processing. |
| Test Credit Card Interface - Card Entry/Display | EOEC_CCI_TEST | Enterprise Components, Component Configurations, Credit Card Interface, Test Credit Card Interface, Card Entry/Display | Enter test credit card information, which you can then submit to verify that your credit card processing is functioning properly. |
| Test Credit Card Interface - Transaction | EOEC_CCI_TRANSACT | Enterprise Components, Component Configurations, Credit Card Interface, Test Credit Card Interface, Transaction | Enter test credit card transaction information, which you can then submit to verify that your credit card processing is functioning properly. |

## Defining Connection Parameters

Access the Credit Card Interface Installation page.

Copyright © 1992-2006, Oracle. All rights reserved.




Credit Card Interface Installation page

Entering information on this page is required for CyberSource, CyberSource-compliant, and XML-based processing systems. Verify connection requirements with your vendor.

**Credit Card Merchant ID** Enter the merchant ID supplied by your vendor.

**Credit Card Hist. Backup Days** (credit card history backup days) If you create a process that archives history records, specify the number of days that you retain credit card authorization history records.

**Credit Card Tracing** Select whether to keep a trace file of credit card transactions. Tracing functionality associated with this field applies to tracing for CyberSource and CyberSource-compliant vendors. Select from the following trace file options:

*Connect with Trace:* Connect to your vendor and create a trace file.

*No Connect with Trace:* Create a trace file only. Do not attempt to connect to your vendor. Use this option to test or troubleshoot credit card transaction data.

*Production:* Send data directly to your vendor without creating a trace file. This is the most efficient method of transmitting your data; however, you have no record of the passed parameters.

*Vendor Trace:* This option is available only for CyberSource. Invoke a troubleshooting utility provided by CyberSource to aid in diagnosing connection problems. Results are displayed in the trace file.

See CyberSource documentation.


Copyright © 1992-2006, Oracle. All rights reserved.

If you trace your activity, the trace file contains a list of the parameters passed for each credit card transaction. If the file already exists, new transactions are appended to the end of the file, providing a running log.

The way in which the trace file is created depends on the setting of the EOEC_CCI_TRANSACTION Business Interlink's trace_file parameter.

If no path and file name are specified in the trace_file parameter, which is the default, a crcard.txt trace file is created in one of the following directories:

- For Windows: c:\temp
- For UNIX: $PS_HOME/appserv/<db_name>

To reconfigure the Credit Card Authorize Bill and Credit integration point to give the trace file a different name or location, modify the EOEC_CCI_TRANSACTION business interlink using PeopleSoft Application Designer.

If only a path is specified for the trace_file parameter, the crcard.txt trace file is created or appended to as defined in the parameter.

If only a file name is specified for the trace_file parameter, the trace file is created or appended to as defined in the parameter in one of the following directories:

- For Windows: c:\temp
- For UNIX: $PS_HOME/appserv/<db_name>

If the path and file name are specified for the trace_file parameter, the trace file is created or appended to as defined in the parameter.

See Chapter 7, "Setting Up the Credit Card Interface," Setting Up Tracing, page 96.

Tracing for an XML-based interface is controlled through PeopleSoft Integration Broker using the Logging field on the routing definition for the service operation.

See *Enterprise PeopleTools 8.48 PeopleBook: PeopleSoft Integration Broker,* "Managing Routing Definitions," Creating Routing Definitions

**On-line Transmission Retries**

Enter a value between *0* and *9* to specify how many times the system should attempt to retransmit transactions in the event of transmission failure.

**Address Verification Flag**

Credit card transmissions can fail authorization if the address you send doesn't exactly match the billing address for the credit card. Select from:

*Add Vf ON* (address verification on): Transactions fail when the address you send does not match the credit card billing address. This is the default value.

*Add Vf OFF* (address verification off): Transactions do not fail when the address you send does not match the billing address on the credit card.

See Chapter 7, "Setting Up the Credit Card Interface," Using the Address Verification Service, page 102.

**Type of Interface**

*Cybersource Compliant:* Select for CyberSource or a CyberSource-compliant vendor. A CyberSource-compliant vendor is one that complies with CyberSource's APIs.

Copyright © 1992-2006, Oracle. All rights reserved. 

*XML Compliant:* Select for an XML-based vendor.

### Allowed Transaction Types

**Credit Card Transaction Type** Select the types of transactions your agents are allowed to submit. Disallowed transaction types are not available on the application-specific credit card transaction page. Select from:

*Authorize Only:* Your vendor verifies that the card is valid for the charge. For example, the customer has enough credit to pay for the order, the card is not stolen, and so forth. The vendor does not bill the credit card.

*Bill Only:* Your vendor bills the card without first verifying that the card is valid for the charge. Select this option if you have pre-authorized the transaction and you want to submit the transaction for billing only.

*Authorize and Bill:* Your vendor performs both authorization and billing during the same transaction. The vendor charges the customer's credit card on receiving authorization.

*Credit Only:* Your vendor credits the customer's credit card.

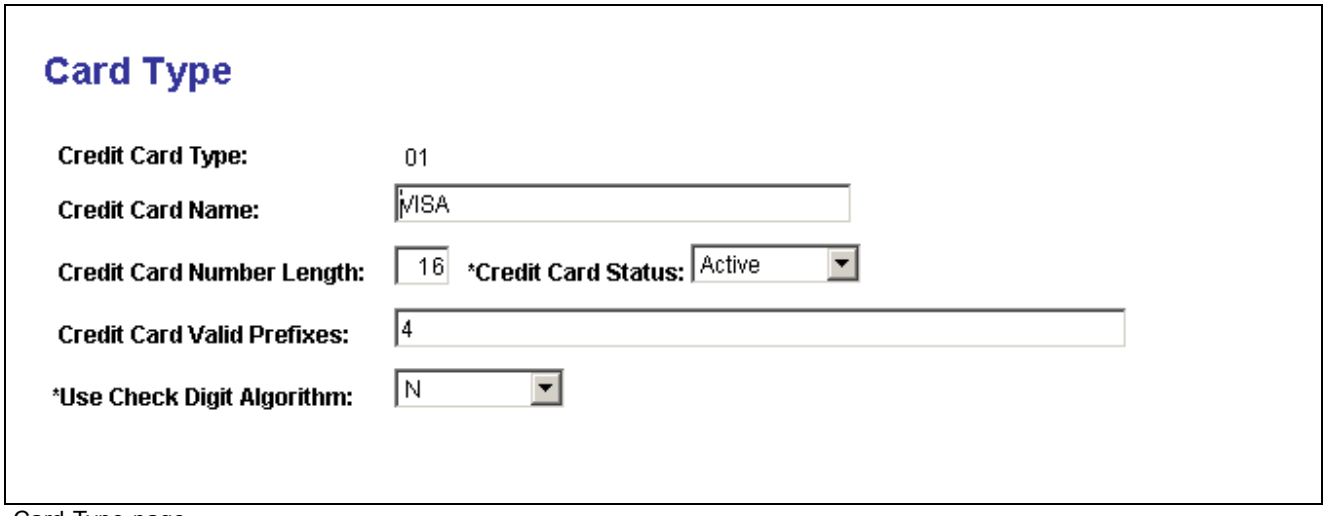
**Process Credits** Select to permit agents to submit credit transactions as well as billing transactions. Available only if you selected either *Authorize and Bill* or *Bill Only* in the Credit Card Transaction Types field.

### Connection Parameters

CyberSource, and CyberSource-compliant and XML-based third-party vendors provide you with information to connect with their systems. Enter that information to enable PeopleSoft to make the connection when you submit a transaction for processing.

## Defining Accepted Credit Card Types

Access the Card Type page.




Card Type page

This page is required for all types of credit card processing.

PeopleSoft delivers data for most popular credit card types. You can modify existing definitions and add new ones.


Copyright © 1992-2006, Oracle. All rights reserved.

**Credit Card Type** The following credit card types are delivered:

*01:* Visa.

*02:* MasterCard.

*03:* Diners Club/Carte Blanche.

*04:* American Express.

*05:* Discover.

**Credit Card Name** Enter a credit card name such as Visa or MasterCard. The name should match the credit card type so that you can identify the card without memorizing the credit card type codes.

**Credit Card Number Length** Enter the card's standard credit card number length. Before transmitting a request to your vendor, the system validates the credit card number length against this number.

**Credit Card Status** Select *Active* if you accept this type of credit card. Select *Inactive* if you don't accept this type of credit card. Inactive credit card types do not appear on the application-specific credit card transaction page nor in the Test Credit Card Interface component.

**Credit Card Valid Prefixes** Enter all valid prefixes for this type of credit card. Enter multiple prefixes in comma-separated format with no spaces in between. The system removes any characters other than numbers and commas when you move to another field.

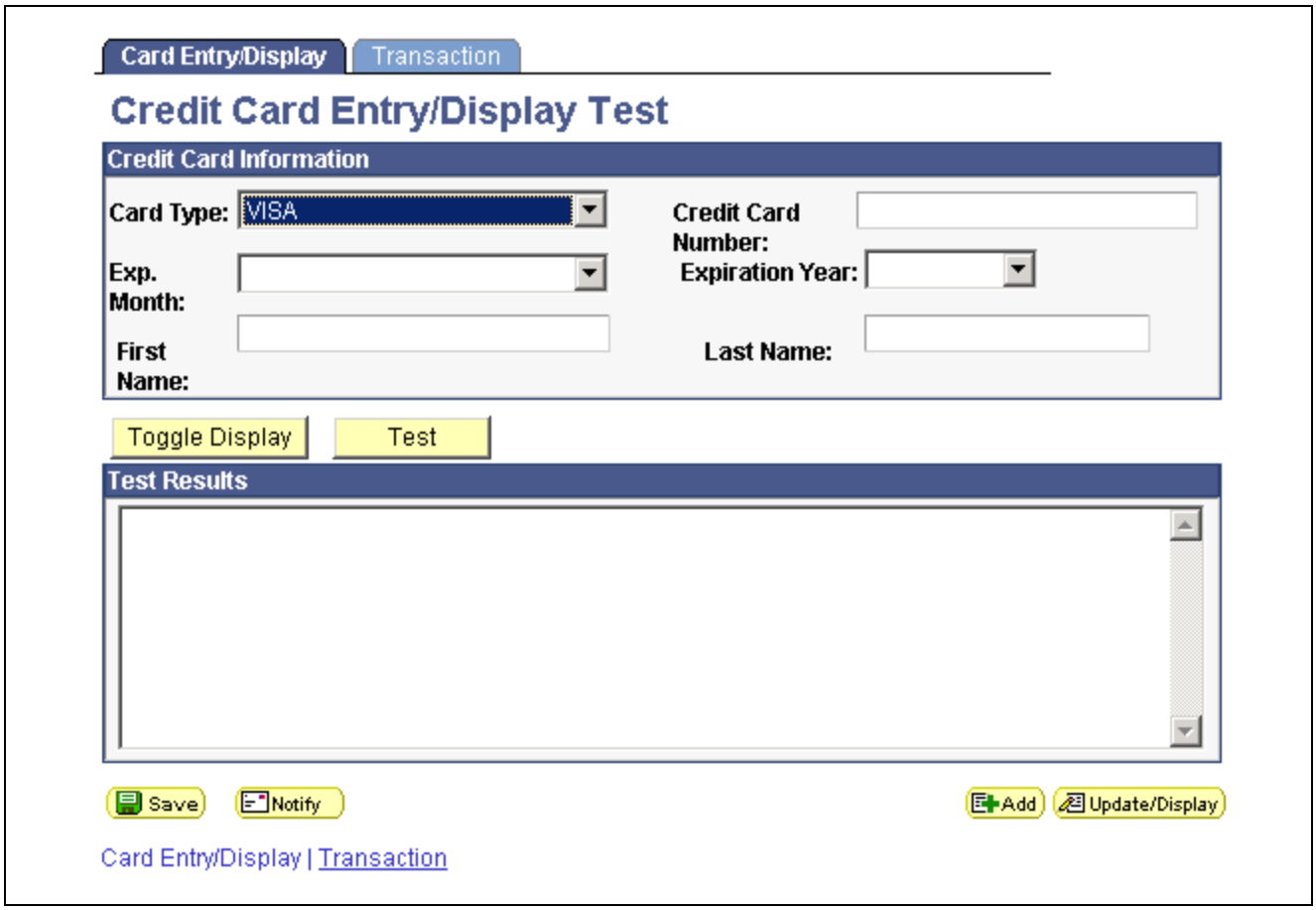
Before transmitting a request to your vendor, the system validates that the credit card number starts with a valid prefix.

**Use Check Digit Algorithm** Select *Y* (yes) to use the modulus (MOD) 10 check digit algorithm to validate credit card numbers before transmitting requests to your vendor. The MOD 10 check digit algorithm verifies whether card numbers you enter into the system are legitimate.

## Testing the Credit Card Interface

Access the Test Credit Card Interface - Card Entry/Display page.

Copyright © 1992-2006, Oracle. All rights reserved. 




Test Credit Card Interface - Card Entry/Display page

The test you can run on this page does the following:

- Verifies that the card number you enter meets the requirements defined in the Credit Card Valid Prefixes field for the associated card type on the Card Type page.
- If you have set the Use Check Digit Algorithm field value to *Y*, verifies that the card number is valid based on the MOD 10 check digit algorithm.
- Verifies that you have entered values in the Exp. Month, Expiration Year, First Name, and Last Name fields on this page.

You can use the following credit card sample data in your test transactions:

| Credit Card Type | Credit Card Number |
|---|---|
| American Express | 378282246310005 |
| Diners Club/Carte Blanche | 38000000000006 |
| Discover | 6011111111111117 |


Copyright © 1992-2006, Oracle. All rights reserved.

| Credit Card Type | Credit Card Number |
|---|---|
| MasterCard | 5555555555554444 |
| Visa | 4111111111111111 |

**Card Type** Select a card type to test. Available values are defined on the Credit Card Types page.

See Chapter 7, "Setting Up the Credit Card Interface," Defining Accepted Credit Card Types, page 82.

**Credit Card Number** Enter the credit card number to test.

**Exp. Month** (expiration month), **Expiration Year, First Name** , and **Last Name** Enter card information to test.

**Toggle Display** Click to switch between display-only and editable modes.

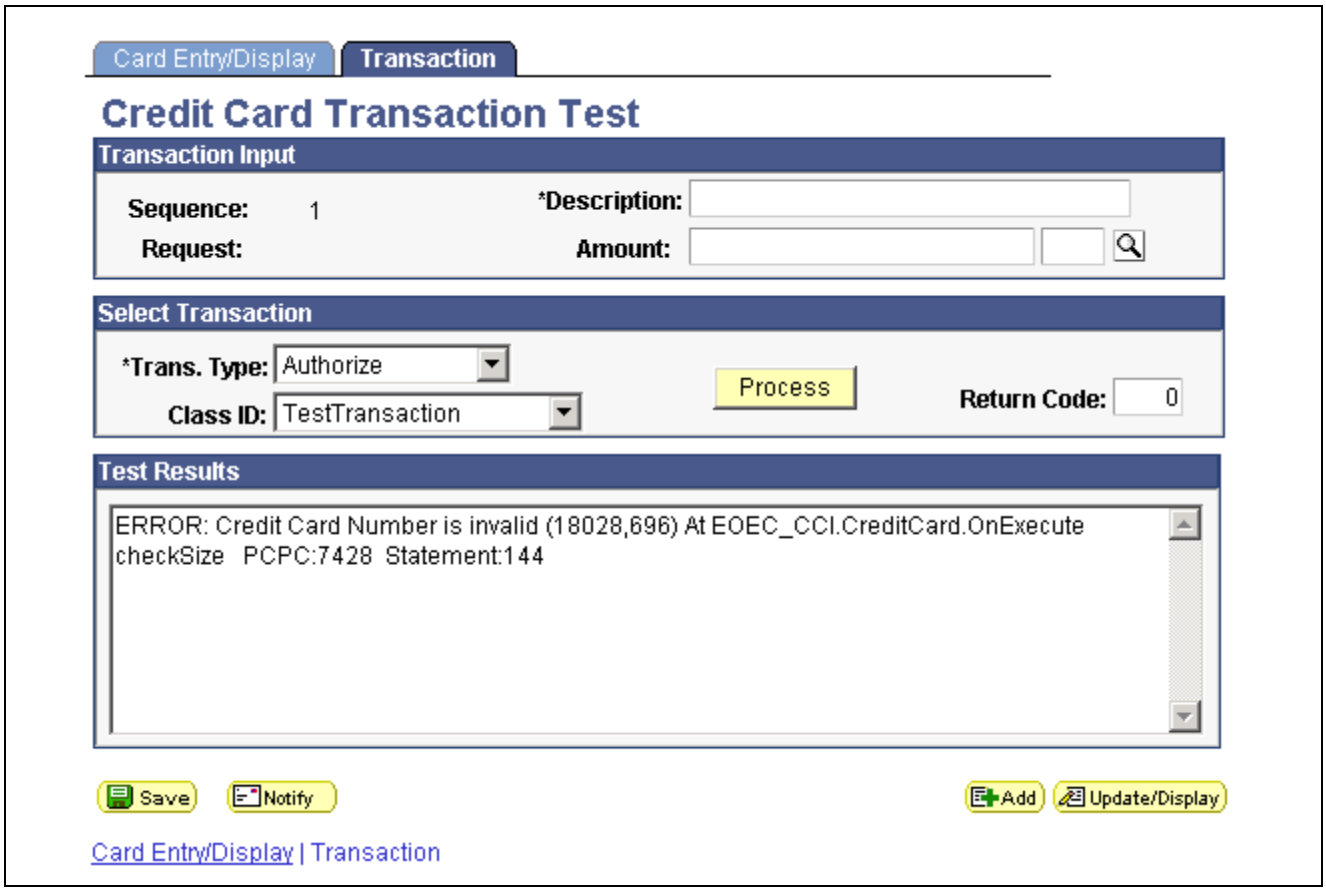
**Test** Click to begin the test.

**Test Results** The results of the test appear in this text box. If the card number is valid, the message VALID CARD NUMBER appears. If the card number is not valid, an explanatory message appears; the card number is incorrect or the card is expired, for example.

## Testing Credit Card Transactions

Access the Credit Card Transaction Test page.

Copyright © 1992-2006, Oracle. All rights reserved.




Credit Card Transaction Test page

**Sequence** and **Request**

Display a combination of numbers that distinguish the transaction from other transactions. They are similar to a run control or job number.

**Amount**

Enter a transaction amount and click the Look up button to select a transaction currency.

**Trans. Type** (transaction type)

Choose a transaction type to test:

*Authorize Only:* Verify that the card is valid for the charge.

*Bill Only:* Bill the card without first verifying that the card is valid for the charge.

*Authorize and Bill:* Perform both authorization and billing during the same transaction.

*Credit Only:* Credit the customer's credit card.

**Class ID**

Select *Test Transaction* (the default) or one of the following interface types that has been specified on the Credit Card Interface Installation page.

*ProcessBrokerTransaction:* Select if you want to test your XML-based interface.

*Interlink Transaction:* Select if you want to test your CyberSource-compliant interface.

See Chapter 7, "Setting Up the Credit Card Interface," Defining Connection Parameters, page 79.


Copyright © 1992-2006, Oracle. All rights reserved.

**Process** Click to process the test transaction.

**Return Code** Enter a return code to test whether proper error messages and results are returned. Available codes and their descriptions are discussed in the Return Codes section below.

**Test Results** The results of the transaction test appear in this text box. Test result interpretations are discussed in the following sections.

## Return Codes

You can enter any of the following return codes and click Process to view the corresponding description and error message in the Test Results area. These return codes and their corresponding errors can appear in multiple areas. For example, when using the Test Credit Card Interface component, they appear on this test page. When using the Business Interlink Tester, they are displayed in the various return fields. In an application, they appear as appropriate to that application's method of interacting with the credit card interface.

| Return Code | Description |
|---|---|
| -3 | Error Opening Trace File |
| -4 | Vendor Error − ICS_INIT failed |
| -5 | Unsupported Service |
| -6 | Credit card number is invalid. |
| -7 | Phone number is too long |
| -8 | State field length is invalid |
| -9 | Zip Code field is too long |
| -10 | Amount must be greater than zero |
| -11 | Vendor Error − ICS_SEND failed |
| -12 | Decryption Failed |
| -15 | Request ID is required |
| -16 | Currency is required |
| -17 | Phone is required |

Copyright © 1992-2006, Oracle. All rights reserved.

| Return Code | Description |
|---|---|
| -18 | Email ID is required |
| -19 | Zip Code is required |
| -20 | City is required |
| -21 | Country code is required |
| -23 | Address 1 is required |
| -99 | Trace Run Only |

# Configuring the Java Interlink Plug-In

PeopleSoft delivers a Java interface to your third-party credit card authorization and payment application.

This section provides an overview of the Java interlink plug-in and discusses how to:

- Set up the CyberSource API and Java plug-in on Microsoft Windows NT systems.
- Set up the CyberSource API and Java plug-in on Solaris systems.

## Understanding the Java Interlink Plug-In

Using PeopleSoft Business Interlink technology, the Java interface supports four types of transactions:

- Authorize only.
- Bill only.
- Authorize and bill.
- Credit.

Use the plug-in to interface with the CyberSource credit card processing services product. The business interlink Java plug-in (delivered with your PeopleTools installation) uses the CyberSource Java API, version 3.7.11. Before you configure your system, obtain the CyberSource Java API from the Support Center in the CyberSource website.

**Note.** This distribution was tested and developed with JDK 1.2.1on Solaris-2.6. If you use a JVM other than that provided by Sun Microsystems, you might encounter problems. This version of the Java SDK does not support JDK 1.1.7.

Download the zipped API files and use the instructions in this section to configure the environment. You must obtain the Java version of the API even if you are already using the C++ plug-in and API.

See http://www.cybersource.com

 Copyright © 1992-2006, Oracle. All rights reserved.

We support CyberSource API version 3.7.11 when used with the following:

- Java Development Kit (JDK) version 1.1.8, 1.2.2, or later; installed and running.
- Solaris 2.6 operating system on SPARC or Windows NT Server 4.0 on Intel.
- If you use a Java Virtual Machine that is not developed and supported by Sun Microsystems, you may encounter problems with the Internet Commerce Suite CyberSource Development Kit for Java.

**Note.** When you upgrade to PeopleTools 8.47 or higher, you must upgrade to Cybersource 3.7.11.

**Note.** Other UNIX platforms are not known to be supported, but please refer to the CyberSource website for a current listing of supported platforms.

# Setting Up the CyberSource API and Java Plug-in on Microsoft Windows NT Systems

This section discusses how to:

- Set up the business interlink environment on Windows NT.
- Set up the CyberSource API on Windows NT.
- Set up the plug-in on Windows NT.

## Setting Up the Business Interlink Environment

To set up the business interlink environment:

1. Ensure that two entries in the PATH environment variable point to the Java Run-Time Environment (JRE) bin.

   Do not confuse the JRE bin with the Java Development Kit (JDK) bin. The two entries in the PATH environment variable should point to the *JRE home*\jre\bin and *JRE home directory*\jre\bin\classic.

2. Create an entry in the CLASSPATH environment variable.

   Point the entry in the CLASSPATH environment variable to the psinterlinks.jar file. The psinterlinks.jar file is located in <*PS_HOME*>\class.

   You must include the name of the .jar file in the CLASSPATH entry; for example, <PS_HOME>\class\psinterlinks.jar.

## Setting Up the CyberSource API

To set up the CyberSource API:

1. Install the CyberSource API using CyberSource instructions.
2. Follow the CyberSource installation directions for making CLASSPATH entries.
3. Update the CyberSource properties file for your CyberSource environment.

   Update the file *CyberSource Home*\ics_3.7.11\properties\ICSClient.props to point to the key and certificate files located in the keys directory. If you are using version 3.7.12, the path is *CyberSource Home*\ics_3.7.12\properties\ICSClient.props.

   Use a forward slash—not a backslash—to separate the directories, as follows:

```
# file with the location of your private key required
```

Copyright © 1992-2006, Oracle. All rights reserved. 

```
myPrivateKey=<CyberSource Home>/keys/<filename.pvt>
# file with the location of your cert required
myCert=<CyberSource Home>/keys/<filename.crt>
# file with the location of the CyberSource cert
serverCert=<CyberSource Home>/keys/CyberSource_SJC_US.crt
```

4. Run the Java class ICSAuthTest.class.

   The Java class ICSAuthTest.class is located in the directory where you installed CyberSource. You must run this to ensure the API is set up properly. To run the ICSAuthTest.class:

   a. Edit the ICSAuthTest.java code to change the expiration to a future date.

   b. Open the Windows command prompt and change drives and directories to point to the test directory.

   c. Type *Javac ICSAuthTest.java* to recompile the edited file.

   d. Type *Java ICSAuthTest* to run the test.

   e. Verify that the Java class ICSAuthTest.class ran successfully.

### Setting Up the Plug-in

To set up the plug-in:

1. Move the pscreditcard.jar file to the desired directory.

   The pscreditcard.jar file is installed in <*PS_HOME*>\class. You can move this .jar file to another directory for more stringent security access. The .jar file operates with execute permission only.

2. Update the CLASSPATH variable to point to the pscreditcard.jar file.

   Include the file name in the CLASSPATH entry.

3. Open PeopleSoft Application Designer, and select File, New, Business Interlink.

4. Select pscreditcard.dll.

5. Verify that the text that appears in the Version field in the new Business Interlink dialog box begins with *plugin.*

   If the text reads *Failed to initialize JVM,* your PATH environment variable is not pointing to the JRE directories. Ensure that your PATH points to both the JRE bin and classic directories; for example: *JRE Home*\jre\bin and *JRE Home*\jre\bin\classic.

6. If you are using three-tier architecture while accessing this plug-in, update the psappsrv.env file.

   To update the psappsrv.env file:

   a. Locate the psappsrv.env file in your application server domain folder (<*PS_HOME*>*appserv**your domain*\psappsrv.env), and open it in a text editor.

   b. Append the path to your JRE \bin\classic directory to the end of the PATH variable in psappsrv.env.

   c. Ensure that there is a semicolon between path entries.

## Setting Up the CyberSource API and the Java Plug-in on Solaris Systems

This section discusses how to:

- Set up the business interlink environment on Solaris.
- Set up the CyberSource API on Solaris.

 Copyright © 1992-2006, Oracle. All rights reserved.

- Set up the plug-in on Solaris.
- Configure the application server on Solaris.

### Setting Up the Business Interlink Environment

To set up the business interlink environment:

1. Ensure that pspkgs.sh was run as part of the installation process.
2. Verify that the file psinterlinks.jar file exists in $PS_HOME/appserv/classes.

   If psinterlinks.jar does not exist, pspkgs.sh may have been run incorrectly.
3. Verify that the Java Virtual Machine is installed under the Tools home directory.

   If the Java Virtual Machine is not installed, pspkgs.sh may have run incorrectly.
4. Locate psconfig.sh, and open it in a text editor.
5. Ensure that there is an entry in the PATH environment variable in psconfig.sh pointing to the JRE bin.

   Do not confuse the JRE bin with the JDK bin. The entry in the PATH environment variable should point as follows: PATH=$PS_JRE/bin:$PATH;export PATH
6. Create an entry in the PS_CLASSPATH environment variable in psconfig.sh.

   Point the entry to the psinterlinks.jar file located in $PS_HOME/appserv/classes. You must include the name of the .jar file in the PS_CLASSPATH entry; for example, $PS_HOME/appserv/classes/psinterlinks.jar.

### Setting Up the CyberSource API

Complete these steps for application servers.

To set up the CyberSource API:

1. Install the CyberSource API using the CyberSource documentation.
2. Update CLASSPATH and PS_CLASSPATH in psconfig.sh to point to the API .jar file. The API .jar file is ics.jar.
3. Update CLASSPATH in psconfig.sh to point to the test directory under the API's ics_3.7.11 or ics_3.7.12 directory.
4. Save your changes to psconfig.sh.
5. Update the CyberSource properties file.

   Update the file *CyberSource Home*/ics_3.7.11/properties/ICSClient.props to point to the key and certificate files located in the keys directory. For version 3.7.12, update the file to CyberSource Home/ics_3.7.12/properties/ICSClient.props.

   Be sure to use a forward slash—not a backslash—to separate the directories, as follows:

   ```
   # file with the location of your private key required
   myPrivateKey=<CyberSource Home>/ics/keys/<filename.pvt>
   # file with the location of your cert required
   myCert=<CyberSource Home>/ics/keys/<filename.crt>
   # file with the location of the CyberSource cert
   serverCert=<CyberSource Home>/ics/keys/CyberSource_SJC_US.crt
   ```
6. Run psconfig.sh to set the properties.

Copyright © 1992-2006, Oracle. All rights reserved. 

7. Run the Java class ICSAuthTest.class.

   The Java class ICSAuthTest.class is located in the directory where you installed CyberSource. You must run this class to ensure that the API is set up properly.

   Before you run the test, edit the ICSAuthTest.java code to change the expiration to a future date and then recompile ICSAuthTest.java by typing *Javac ICSAuthTest.java* at the prompt.

---

**Note.** You do not need to make a system-level CLASSPATH entry in addition to the CLASSPATH entry in psconfig.sh. If the Java class ICSAuthTest.class runs successfully, you can remove the CLASSPATH references to the test directory and the ics.jar file.

---

### Setting Up the Plug-in

Complete these steps for application servers and PeopleSoft Process Scheduler servers.

To set up the plug-in:

1. Confirm that the pscreditcard.jar file is located in *$PS_HOME*/appserv/classes.

   You can move this .jar file to another directory to tighten security and limit access. If you move this .jar file to a different location, ensure that you reference the new location in the following steps.

2. Update PS_CLASSPATH in psconfig.sh to point to the pscreditcard.jar file in the classes directory.

   Include the file name in the entry.

3. Confirm that the libpscreditcard file is in the *$PS_HOME*/bin/InterfaceDrivers directory.

   ---

   **Note.** Use the same naming convention as the libpsjavaproxy file for the libpscreditcard file.

   ---

4. Stop and restart the application server.

   Stop the services; do not stop the machine.

### Configuring the Application Server

To configure the application server:

1. Copy the ICSClient.props file from the CyberSource properties directory to the application server domain directory.
2. Open the psappsrv.env text file that is located in your application server (not the web server) domain folder.
3. Append the path to your JRE/bin/classic directory to the end of the PATH variable.

   Ensure that there is a semicolon between path entries. You can find the JRE path in your PATH system variable.

4. Shut down and restart your application server.

# Configuring the PeopleSoft Application Designer Plug-in

This section discusses how to:

- Configure access to account information.

 Copyright © 1992-2006, Oracle. All rights reserved.

- Configure proxy server support.
- Set up tracing.
- Check parameter check logic.
- Set the decryption option.

---

**Note.** All credit card transactions are supported through one business interlink transaction.

---

# Configuring Access to Account Information

The following configuration parameters are needed to complete any transaction:

- Merchant ID
- Private key file
- Certificate file
- Server ID
- Server certificate
- Server URL

The plug-in offers the following three configuration options:

- Maintain the six configuration parameters in a properties file.

  Use this option if you are building a new credit card processing application.

- Pass the merchant ID and server URL as input parameters, and specify the path of the key and certificate files.

  Use this option if you have an existing application that has used the C++ plug-in, but which you are now porting to Solaris.

- Use this plug-in just as you use the C++ version.

  The plug-in gets the certificate and key files from <CyberSource Home>\keys. Use this option if you have been using the C++ plug-in and want to continue to operate on a Microsoft Windows platform.

---

**Note.** The following configuration options demonstrate default input parameter values. After setting up a transaction, set the input parameter values as default values to test them in the Business Interlink Tester. Typically, PeopleCode sets these input parameter values at runtime.

In PeopleSoft Application Designer, select File, Open, Business Interlink to open a business interlink transaction. Define the configuration parameters for your credit card business interlink definition on the Settings and Input pages.

---

## Option 1: Keep All Configuration Parameters in a Properties File

Use this option if you are building a new credit card processing application.

To keep all of the configuration parameters in a properties file:

1. Update the CyberSource properties file.

   Update the file <CyberSource Home>\ics_3.7.11\properties\ICSClient.props to point to the key and certificate files located in the keys directory. If you are using version 3.7.12, update the file to Update the file <CyberSource Home>\ics_3.7.12\properties\ICSClient.props

   Use a forward slash to separate directories, as follows:

Copyright © 1992-2006, Oracle. All rights reserved. 

```
# file with the location of your private key required
myPrivateKey=<CyberSource Home>/keys/<filename.pvt>
# file with the location of your cert required
myCert=<CyberSource Home>/keys/<filename.crt>
# file with the location of the CyberSource cert
serverCert=<CyberSource Home>/keys/CyberSource_SJC_US.crt
```

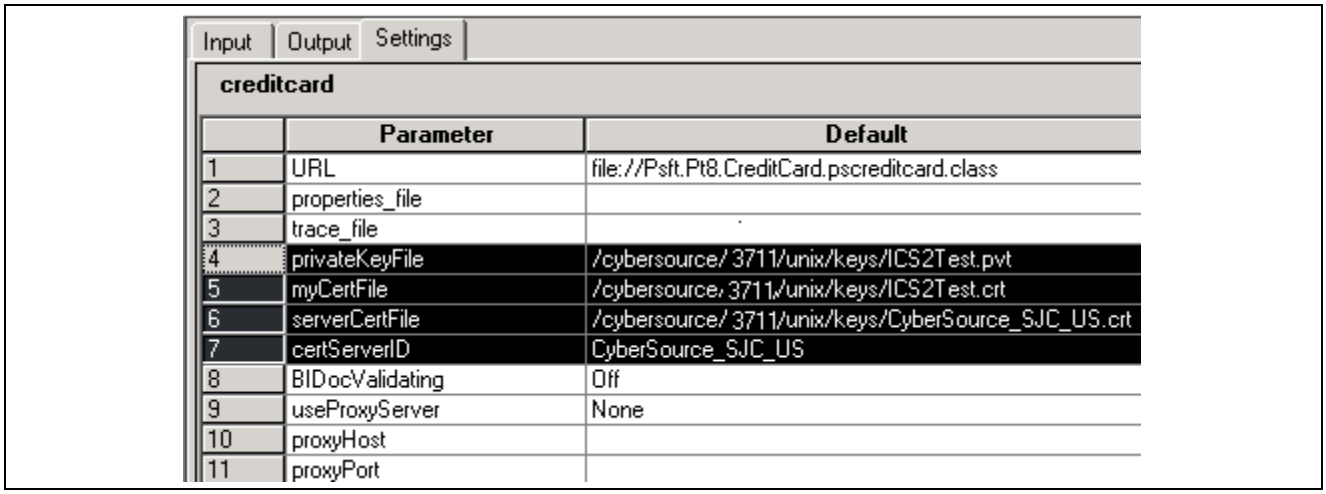
2. Set the value of the properties_file configuration parameter to the path of the ICSClient.props file.

   In PeopleSoft Application Designer, select the Settings tab to set the value. Use a forward slash to separate directories.

3. Leave the privateKeyFile, myCertFile, serverCertFile, and certServerID fields blank.

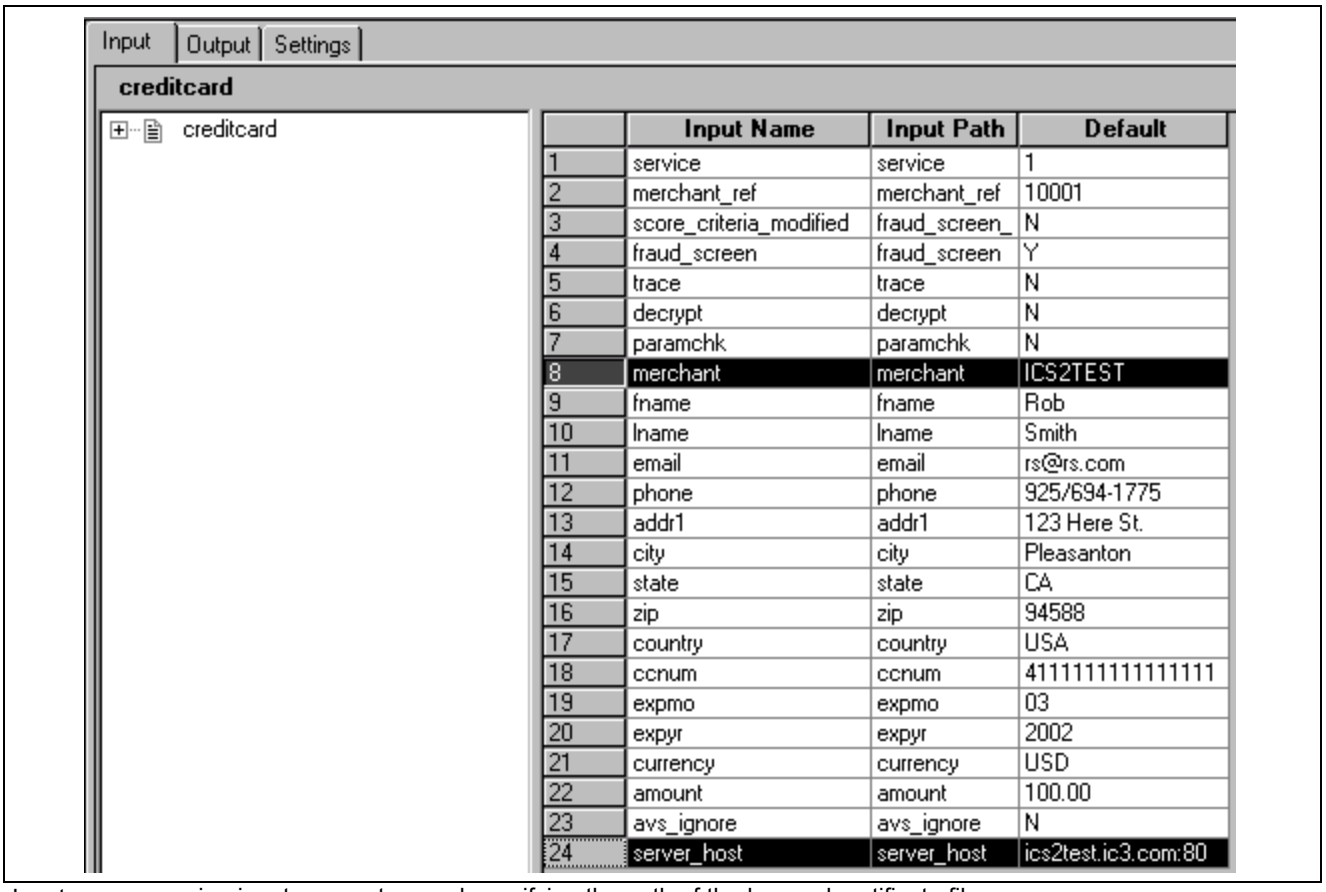
### Option 2: Pass the Merchant ID and Server URL as Input Parameters, and Specify the Path of the Key and Certificate Files

Use this option if you have an existing application that has used the C++ plug-in but which you are now porting to Solaris.




Settings page used to pass input parameters and specify the path of the key and certificate files

 Copyright © 1992-2006, Oracle. All rights reserved.




| | Input Name | Input Path | Default |
|---|---|---|---|
| 1 | service | service | 1 |
| 2 | merchant_ref | merchant_ref | 10001 |
| 3 | score_criteria_modified | fraud_screen_ | N |
| 4 | fraud_screen | fraud_screen | Y |
| 5 | trace | trace | N |
| 6 | decrypt | decrypt | N |
| 7 | paramchk | paramchk | N |
| 8 | merchant | merchant | ICS2TEST |
| 9 | fname | fname | Rob |
| 10 | lname | lname | Smith |
| 11 | email | email | rs@rs.com |
| 12 | phone | phone | 925/694-1775 |
| 13 | addr1 | addr1 | 123 Here St. |
| 14 | city | city | Pleasanton |
| 15 | state | state | CA |
| 16 | zip | zip | 94588 |
| 17 | country | country | USA |
| 18 | ccnum | ccnum | 4111111111111111 |
| 19 | expmo | expmo | 03 |
| 20 | expyr | expyr | 2002 |
| 21 | currency | currency | USD |
| 22 | amount | amount | 100.00 |
| 23 | avs_ignore | avs_ignore | N |
| 24 | server_host | server_host | ics2test.ic3.com:80 |

Input page - passing input parameters and specifying the path of the key and certificate files

To pass the merchant ID and server URL as input parameters and to specify the path of the key and certificate files:

1. Delete the value of the properties_file configuration parameter on the Settings page.

   Enter the paths for the key and certificate files in their appropriate configuration parameters. Use a forward slash to separate directories. The default value for the certificate server ID is *CyberSource_SJC_US*.

2. Include the merchant and server_host input parameters in your transaction.

   Do not include the text *http://* in the server URL string.

## Option 3: Use the Java Plug-in as You Use the C++ Version

Use this option if you have been using the C++ plug-in and want to continue to operate on a Windows platform.

To use the Java plug-in as you use the C++ version:

1. Delete the values of the properties_file, certificate, key, and server ID field parameters on the Settings page of the business interlink definition.
2. Use the merchant and server_host field input parameters in your transaction.

   Do not include the text *http://* in the server URL string.

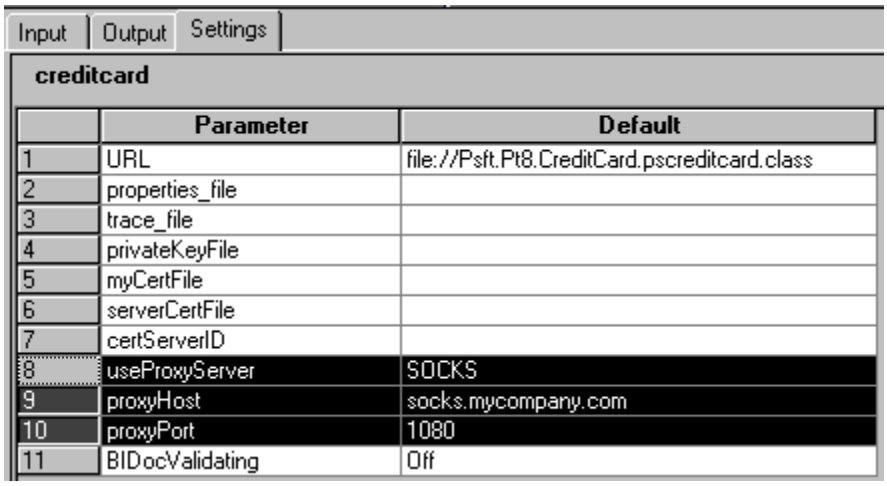
Copyright © 1992-2006, Oracle. All rights reserved.

# Configuring Proxy Server Support

The Java plug-in supports the HTTP proxy without a user name and password, and it supports the SOCKS proxy protocols.

See CyberSource ICS2 Developer's Guide

The following example shows the settings for configuring proxy support:




Settings page for implementing proxy server support

To implement proxy server support:

1. Select the proxy type from the useProxyServer configuration parameter.
2. Enter the values of the host and port in the proxyHost and proxyPort configuration parameters.

# Setting Up Tracing

The plug-in and the CyberSource API support the use of a trace file. To keep a log of the transactions, use the plug-in's trace file and reserve the API's trace file for debugging. The plug-in trace file:

- Uses the input names as they appear in PeopleSoft Application Designer, not CyberSource API property names.
- Prints only the last four digits of the credit card number.
- Prints output values and any error messages.

## Activating Tracing

There are two methods for activating tracing in the Java plug-in:

- Leave the trace_file blank and use the trace input parameter to toggle tracing.

  The trace file writes to c:\temp\crcard.txt.

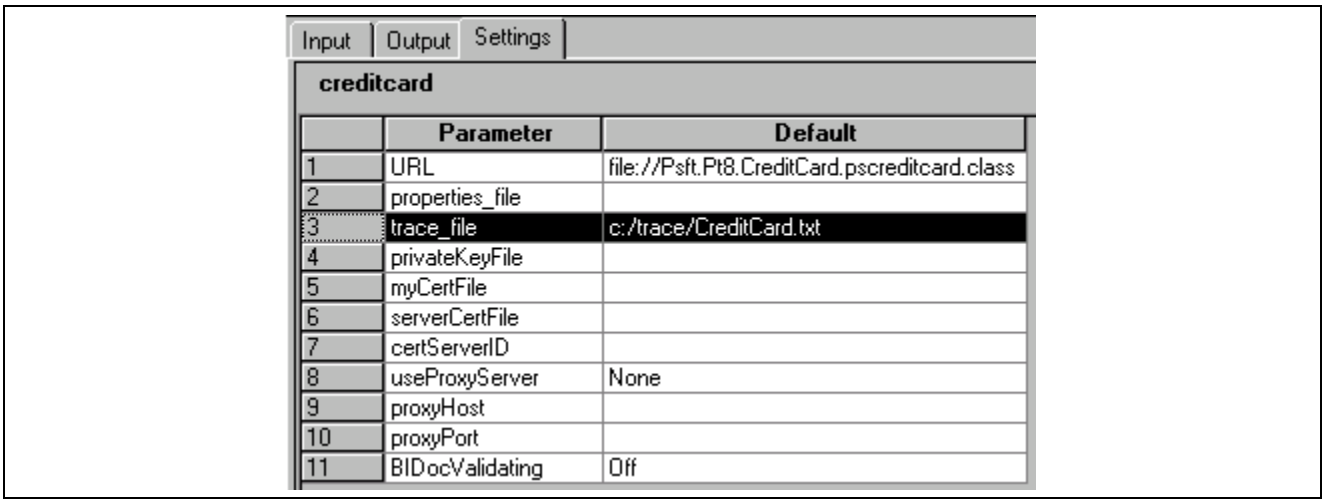
- Enter a path and file name in the trace_file configuration parameter, and use the trace input parameter to enable the tracing (trace=*Y*) or disable the tracing (trace =*N)*.

  Use this method for new credit card definitions, for porting existing credit card definitions to Solaris, or for writing the trace file to a directory other than c:\temp. Use forward slashes when specifying directory paths.

  See Chapter 7, "Setting Up the Credit Card Interface," Defining Connection Parameters, page 79.

 Copyright © 1992-2006, Oracle. All rights reserved.

The following screen shot illustrates the second method.




Settings page used to activate tracing

### Activating Trace Run-Only Mode

To see the input values that are sent to the CyberSource API and to confirm that the plug-in can instantiate the CyberSource interface without sending the transaction to CyberSource, pass a value of *T* with the trace input parameter. If a trace file has been specified in the trace_file configuration parameter, the system uses that trace file. Otherwise, the system uses the c:\temp\crcard.txt trace file.

## Setting Parameter Check Logic

If the paramchk parameter is set to *Y,* the plug-in runs in trace run-only mode and performs additional checks in the input parameters to ensure that the parameters do not exceed maximum allowed field lengths. The system sends the transaction information to the CyberSource API.

## Setting the Decryption Option

Credit card numbers can be sent encrypted to the plug-in. Do not change this option unless you also change the PeopleCode to pass the credit card number as plain text.

# Setting Transaction Inputs and Outputs

If you create a new business interlink definition, you need to set inputs and outputs.

This section provides an overview of required parameters for available transactions and discusses how to:

- Use authorize-and-bill transactions.
- Use authorize-only transactions.
- Use bill-only transactions.
- Use credit transactions.
- Use the address verification service.
- Use the fraud screen service.

Copyright © 1992-2006, Oracle. All rights reserved. 

- Identify input fields that are not used or supported.

# Identifying Required Parameters

The service and merchant_ref input parameters are required in all credit card transactions. The method that you use to configure the plug-in determines which fields are required.

The service and merchant_ref input parameters are required in all credit card transactions. The method that you use to configure the plug-in determines which fields are required.

## Service Input Field

The system uses the service parameter to determine which of the four credit card transactions it should conduct through business interlink transactions. The service parameter values perform the following functions:

| Value | Function |
|---|---|
| 1 | Authorize Only |
| 2 | Authorize and Bill |
| 3 | Bill Only |
| 4 | Credit |

## Merchant_ref Input Field

The merchant_ref field is a merchant-generated order reference or tracking number, which contains information such as an order number. Use the merchant_ref_number field as your own tracking number for the order, to keep track of requests sent to CyberSource relating to the same order.

# Using Authorize-and-Bill Transactions

The authorize-and-bill transaction (service=*2)* performs both authorize and bill actions and returns a bill amount for verification of the amount billed.

Required input data includes:

- Service
- Merchant_ref
- Fname
- Lname
- Addr1
- City
- Country
- State
- Zip

 Copyright © 1992-2006, Oracle. All rights reserved.

- Email
- Expmo
- Expyr
- Ccnum
- Amount
- Currency

Output data includes:

- return_status
- Return_status_msg
- Ret_msg1
- Ret_msg2
- Ret_authcd
- Ret_authdttm
- Rqstid_out
- Bill_amount

## Using Authorize-Only Transactions

The authorize-only transaction (service=*1)* authorizes the amount passed in and returns a request ID to be used later in a bill transaction.

Required input data includes:

- Service
- Merchant_ref
- Fname
- Lname
- Addr1
- City
- Country
- State
- Zip
- Email
- Expmo
- Expyr
- Ccnum
- Amount
- Currency

Output data includes:

Copyright © 1992-2006, Oracle. All rights reserved.

- Return_status
- Return_status_msg
- Ret_msg1
- Ret_msg2
- Ret_authcd
- Ret_authdttm
- Rqstid_out

# Using Bill-Only Transactions

The bill-only transaction (service=*3)* requires the use of a request ID obtained from an authorize-only transaction.

Required input data includes:

- Service
- Merchant_ref
- Amount
- Currency
- Rqstid

Output data includes:

- Return_status
- Return_status_msg
- Ret_msg1
- Ret_msg2
- Ret_authcd
- Ret_authdttm
- Rqstid_out
- Bill_amount
- Trans_ref_no

# Using Credit Transactions

You can use one of the following methods to perform a credit transaction.

## Use a Request ID

This option enables you to use the value of a request ID returned from a previous request for bill transactions. This value is matched with a previous bill for the same order. If this field is present and there is no matching bill, the transaction fails. If the bill request ID is not present, the credit transaction requires the customer billing information.

Required input data includes

- Service

 Copyright © 1992-2006, Oracle. All rights reserved.

- Merchant_ref
- Amount
- Currency
- Rqstid

Output data includes:

- Return_status
- Return_status_msg
- Ret_msg1
- Ret_msg2
- Ret_authcd
- Ret_authdttm
- Rqstid_out
- Credit_amount
- Trans_ref_no

### Use Billing Information

Required input data includes:

- Service
- Merchant_ref
- Fname
- Lname
- Addr1
- City
- Country
- State
- Zip
- Email
- Phone
- Expmo
- Expyr
- Ccnum
- Amount
- Currency

Output data includes:

- Return_status
- Return_status_msg

Copyright © 1992-2006, Oracle. All rights reserved.

- Ret_msg1
- Ret_msg2
- Ret_authcd
- Ret_authdttm
- Rqstid_out
- Credit_amount

## Using the Address Verification Service

CyberSource returns the decline code *DAVSNO* if the billing address does not match the billing address for the credit card. If you set the value of the avs_ignore field to *yes* and include it in your business interlink definition, the system allows the authorization transaction to proceed despite the billing address discrepancy. If the value of the avs_ignore field is anything other than *yes,* the address verification proceeds as usual.

The address verification service returns a value that is stored in the ret_avs field. The values are:

| Value | Description |
|---|---|
| A | The street number matches, but the 5-digit ZIP code and 9-digit ZIP code do not match. |
| E | Address verification data is invalid. |
| N | The street number, 5-digit ZIP code, and 9-digit ZIP code do not match. |
| R | System unavailable. |
| S | Issuer does not offer address verification. |
| U | Address information unavailable. This code occurs if a customer tries to use an international card, or if the address verification in United States banks is not functioning properly. |
| W | The street number does not match, but the 5-digit ZIP code and 9-digit ZIP code match. |
| X | Exact match. The street number, 5-digit ZIP code, and 9-digit ZIP code match. |
| Y | The street number and 5-digit ZIP code match. |
| Z | The 5-digit ZIP code matches. |


Copyright © 1992-2006, Oracle. All rights reserved.

## Using the Fraud Screen Service

Both the authorize-only and the authorize-and-bill transactions support the fraud screen transaction.

See CyberSource documentation.

The Fraud_screen field determines if the fraud screen transaction request should be sent with the current transaction.

The Score_criteria_modified field determines if fraud screen score tuning parameters were altered for this transaction. If this field is set to *yes,* all of the following score tuning parameter fields must be included in the transaction.

Required data input includes:

- Service
- Merchant_ref
- Fname
- Lname
- Addr1
- City
- Country
- State
- Zip
- Email
- Phone
- Expmo
- Expyr
- Ccnum
- Amount
- Currency

Output data includes:

- Return_status
- Return_status_msg
- Ret_msg1
- Ret_msg2
- Ret_authcd
- Ret_authdttm
- Rqstid_out
- Bill_amount (authorize/bill transaction)

Copyright © 1992-2006, Oracle. All rights reserved. 

# Identifying Input Fields That Are Not Used or Supported

The following input fields are not used or supported as of 11/06/00:

| Input Field | Comment |
|---|---|
| Cctype | Not used by CyberSource. |
| Ip_address | Proxy server support is included. The Java CDK supports use of both HTTP and SOCKS proxy |
| Extra1 | N/A |
| Extra2 | N/A |


Copyright © 1992-2006, Oracle. All rights reserved.

# PART 4

# PeopleSoft Active Analytics Framework

**Chapter 8**
**Understanding PeopleSoft Active Analytics Framework**

**Chapter 9**
**Building and Managing Policies**

**Chapter 10**
**Setting Up the Data Library**

**Chapter 11**
**Managing Contexts**

**Chapter 12**
**Setting Up the Action Framework**

**Chapter 13**
**Administering the Framework**

**Chapter 14**
**Considerations for Enabling the Framework**

# CHAPTER 8

# Understanding PeopleSoft Active Analytics Framework

This chapter provides an overview of PeopleSoft Active Analytics Framework and discusses:

- Policies and trigger points.
- The data library.
- The action framework.

## Understanding PeopleSoft Active Analytics Framework

PeopleSoft Active Analytics Framework is a suite of tools comprising a closed-loop decision-making system where business-intelligent applications or transactions can respond when conditions are met and specific actions are recommended; for example:

- Giving a priority service or a better discount for high-value customers
- Send pertinent emails or notifications.
- Displaying alerts and warning messages.

PeopleSoft Active Analytics Framework provides components for setting up the analytic framework, which include managing the data library, building policies, and managing trigger points and actions. These components provide a way to define flexible business rules, called policies, that can be altered without modifying application code. Business analysts and other functional users define policies with an intuitive user interface.

Functional users can create policies that use data elements of various forms and shapes residing in different sources such as the transactional environment, data warehouses, legacy systems, and so on forth. The data elements are exposed to the business user as terms defined in the PeopleSoft Active Analytics Framework data library.

The extensible action framework supports the definition and execution of consequential actions. Application developers can create customized action types within product lines to accommodate their functional needs. PeopleSoft also delivers a built-in action type for displaying alerts to the user.

## Understanding Policies and Trigger Points

Policies are the result of combining trigger points, conditions, and defined actions to complete a desired business request. The framework includes components for building policies.

Copyright © 1992-2006, Oracle. All rights reserved.

Application developers and functional business analysts use a wizard-like interface to build, manage, and associate trigger points to policies. Business analysts can create interactive policies that react to customer behavior of particular interest to them. For example:

- If a banking customer deposits more than ten thousand dollars, a banking analyst can create a regulatory I.R.S. notification.
- If a high-value customer has logged three or more critical support issues with a call-center within a week, a business executive could send a letter of apology.

Policies supplement, but do not alter, normal transactional processing. Therefore, a policy cannot be used to stop a particular behavior due to some specified restriction. If the condition portion of a policy evaluates to true, the policy causes an action to be performed. If the specified condition evaluates to false, then no consequential actions occur. Policies are evaluated independently from each other. Therefore, if more than one policy is to be evaluated at the same time, the consequential actions of one policy cannot alter or influence the actions of another. Likewise, the sequence of policy execution cannot affect the results of another policy.

You construct a policy by defining one or more conditions, specifying one or more actions, and associating them to a trigger point. A policy cannot be activated without defining at least one condition and action. You can reuse defined policies with multiple trigger points if the elements of the policy agree with the contexts of the trigger points.

*Trigger points* are events from which the analytic decision engine is invoked by the application. The framework supports the registration of new trigger points, when needed. Examples of trigger points are:

- When a customer is identified.
- When a product is selected.
- After logging in to a self-service application.

Registration of a trigger point involves specifying:

- The type of actions to be invoked.
- The *context* in which the associated policies are to be executed.

During runtime, policies are triggered by specific trigger points within application components, resulting in defined actions being taken.

---

**Note.** Registration of a trigger point also involves introducing necessary code in the application to request the decision engine to evaluate the policies pertaining to a trigger point.

---

# Understanding the Data Library

The data library is a repository for information within the PeopleSoft Active Analytics Framework. Each element in the data library is exposed by way of a *term*, which is a pointer to a unit of data within the PeopleSoft system. This data may reside in a relational database, or it may be derived at runtime.

Terms in the data library are organized hierarchically into functional categories called *subject areas*. Terms can be assigned to more than one subject area at a time; for example, if a term represents a customer it could be located both in a marketing subject area and in a financial subject area.

The data library enables functional users to:

- Access data (terms) residing in different sources such as an operational CRM environment, data warehouses, legacy systems and so on.

 Copyright © 1992-2006, Oracle. All rights reserved.

- Use the data in variety of contexts, such as input in rule applications, tokens in correspondence templates, or placeholders for customized text in questions or for presenting disparate pieces of information in different screen applications.

PeopleSoft Active Analytics Framework includes components to define new terms in the data library and can automatically create terms for data elements in a component.

# Understanding the Action Framework

The action framework is a suite of components that are designed to manage actions and to invoke actions at runtime. The primary purpose of the action framework is to allow functional users to specify and configure the actions to be performed when policy conditions evaluate to true.

Also, PeopleSoft designed the action framework to enable application developers and IT personnel to introduce new action types for use in policies and to invoke them at runtime.

The action framework components:

- Register and maintain *action types*. An action type is metadata pertaining to a class of actions that can be performed at runtime.
- Register and maintain *action bundles*. Action bundles are groups of combinable action types.
- Embed and configure consequential actions within policies.
- Provide a display alert action type for use with all product lines.

Copyright © 1992-2006, Oracle. All rights reserved.

 Copyright © 1992-2006, Oracle. All rights reserved.

# CHAPTER 9

# Building and Managing Policies

This chapter discusses how to:

- Build policies.
- Manage Trigger Points.

## Building Policies

You build and manage policies with a wizard-like interface called a policy builder. During the creation of policies, you associate them to trigger points. At runtime, policies are evaluated by specific trigger points in application components, resulting in defined actions being taken.

The policy builder allows business analysts to change conditions or actions or both in policies to enable a business process change in an application component, without having to modify application code, or needing the help of IT personnel.

**Note.** Policies cannot be shared among different setIDs.

### Prerequisites to Building Policies

Before building a policy, you should have already:

- Defined trigger points.
- Defined data library terms.
- Defined action types, categories, and action objectives.

See Chapter 13, "Administering the Framework," Registering Trigger Types and Trigger Points, page 154.

See Chapter 10, "Setting Up the Data Library," Creating Terms, page 130.

See Chapter 12, "Setting Up the Action Framework," page 143.

Copyright © 1992-2006, Oracle. All rights reserved.

## Pages Used in This Section

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Manage Policies | EOCF_RULE_CFGSRCH | Enterprise Components, Active Analytics Framework, Policies, Manage Policies. | Used to build and manage policies. |
| Manage Trigger Point | EOCF_MANAGE_TP | Enterprise Components, Active Analytics Framework, Policies, Manage Trigger Point | Used to manage trigger points. |

## Building, Editing and Activating Policies

To build a new policy or edit an existing policy, access the Manage Policy page. If you want to edit an existing policy, use the search criteria to find the desired policy, then click the policy name on the results grid to open the policy definition.

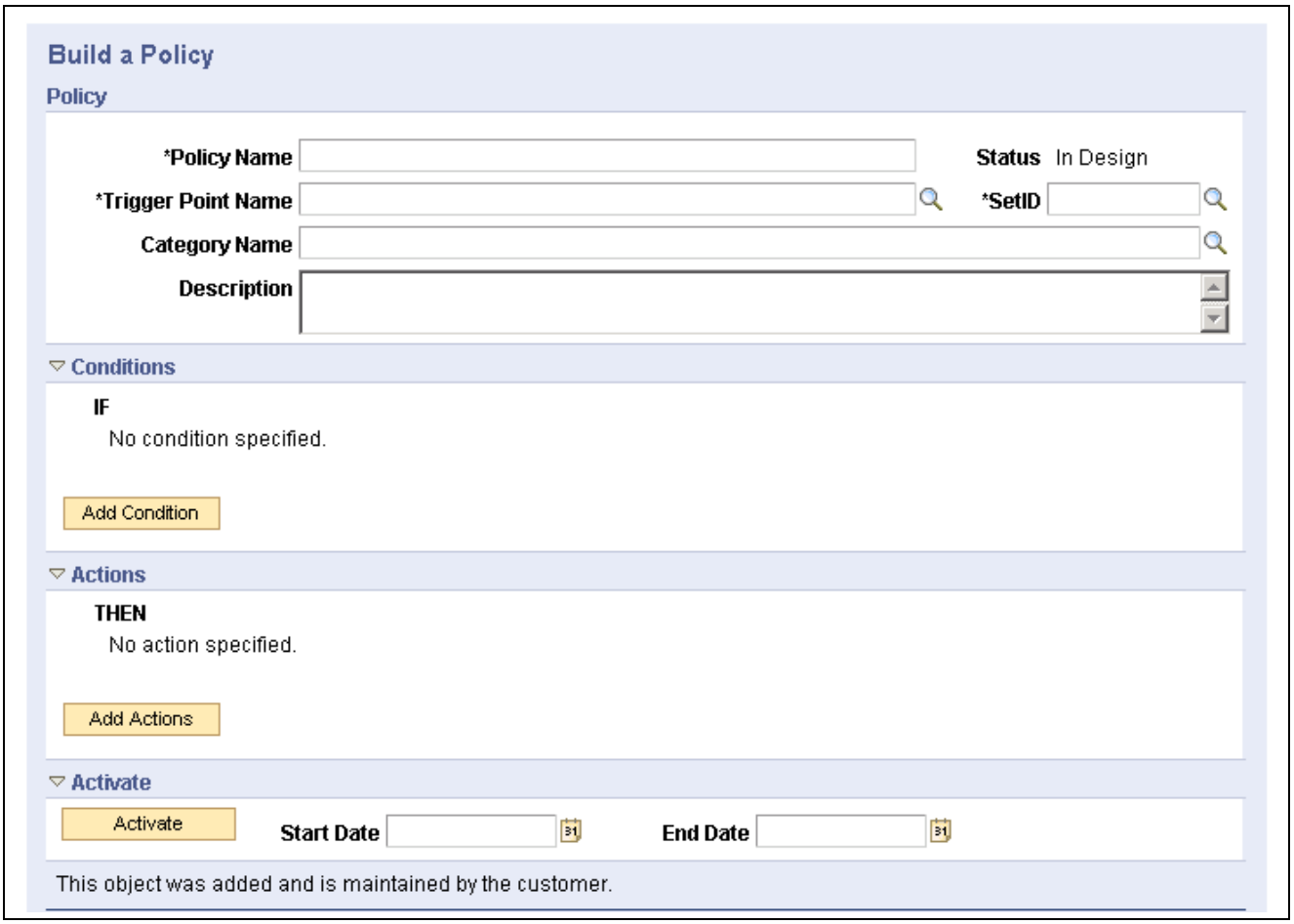
---

**Note.** If you select a trigger point name as a search criterion, only policies directly associated to the trigger point are retrieved. To retrieve policies associated to *policy groups* of a trigger point, specify search criteria other than the trigger point name.

---

To create a new policy, click Build Policy.


Copyright © 1992-2006, Oracle. All rights reserved.




Build Policy page

| | |
|---|---|
| **Policy Name** | Unique and descriptive name of a policy. |
| **Trigger Point Name** | Descriptive name of a trigger point. A policy is always associated to at least one trigger point. |
| **Category Name** | Descriptive name of a policy category, used to functionally classify policies and aid in searching for policies. |
| **SetID** | Defined within the PeopleSoft system. Specify a setID and a trigger point for every policy. This value is used to select the valid set of policies associated to a trigger point at runtime; it also constrains the prompt list for the right-hand side values entered in conditions by performing a setID to setID indirection. |
| **Status** | This value defaults to In Design when the policy is new. Activation of a policy changes the status to Active. |
| **Start Date, End Date** | Dates defining the validity of a policy at runtime. |

Enter the appropriate fields and click Add Conditions.

## Adding and Editing Conditions

Click Add Condition on the Build Policy page.

There are two modes of specifying conditions:

Copyright © 1992-2006, Oracle. All rights reserved. 

- Basic.

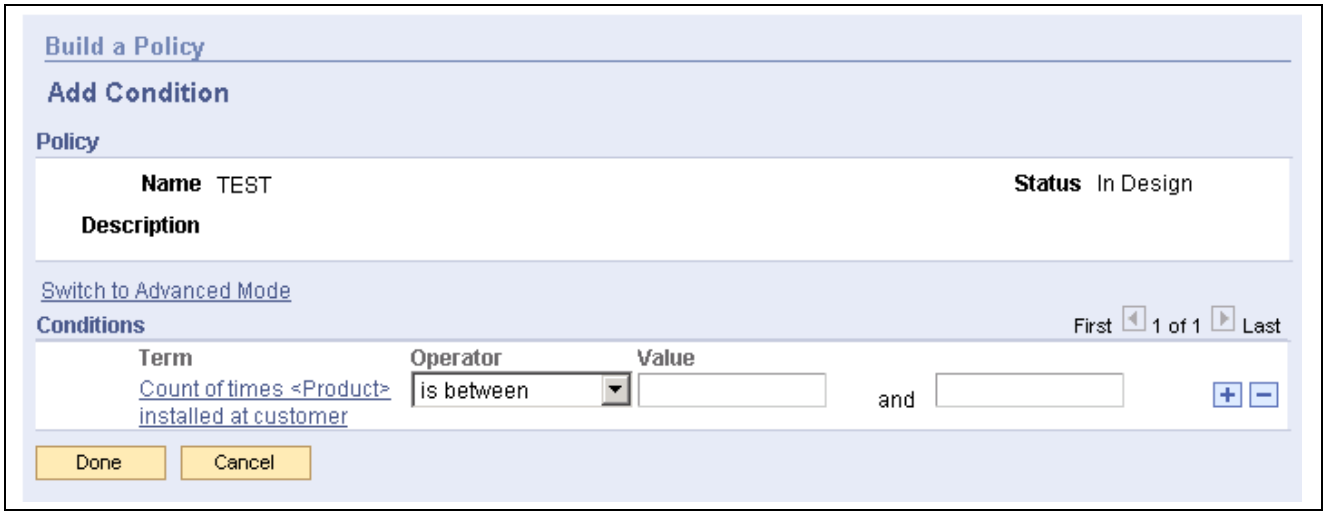
  This is the default mode; the logical operator defaults to AND.

- Advanced.

  You can group condition rows using parenthesis, specify logical operators (AND, OR), and specify terms as values in the right-hand side.

To add a condition row:

1. Select a term and, if it is hyperlinked, configure it.
2. Select an operator.
3. Enter or select one or more values on the right hand side.




Build a Policy page

## Selecting Available Terms

Click Select Term on the Add Condition page to display the Term Selection page.

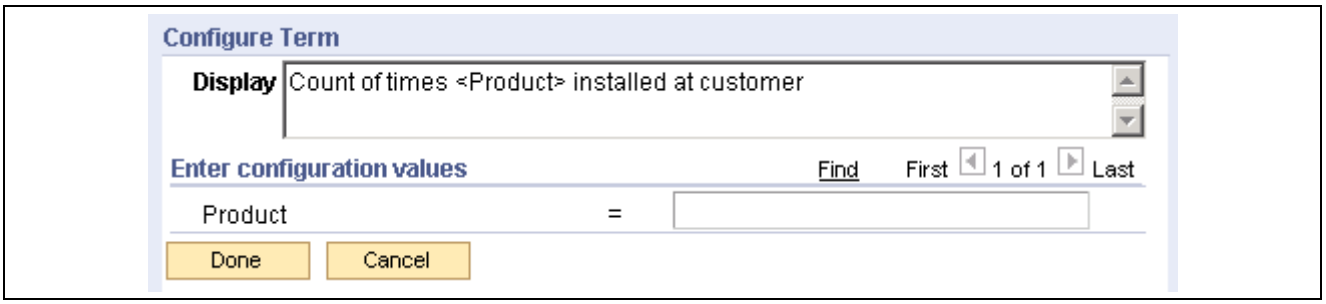
The Term Selection page has two modes in which to search and select terms for a condition: browsing by subject area or searching by entering search criteria. Terms that appear are limited to those that can be resolved by the trigger point. Therefore, all terms having contextual implementations for the context pertaining to this trigger point, and all terms having generic implementations in which binds can be supplied by the context will be available.

Terms returning multiple rows are not available for use in conditions. Terms that retrieve data from detail rows in the component buffer cause the decision engine to evaluate the condition once for each row. The decision engine generates actions for every row for which the condition is true.

## Configuring a Term

If you select a term that is configurable; that is, it has design time binds, it must be configured when you build the condition. Configurable terms are displayed as hyperlinks. Subsequently, while building the condition, click this hyperlink to configure the term.


Copyright © 1992-2006, Oracle. All rights reserved.




Configure Term area

The prompt of valid values displayed for configurable terms relies on the prompt configuration specified in the Manage Terms component.

See Chapter 10, "Setting Up the Data Library," Creating Terms, page 130.

## Selecting an Operator

The list of operators available when you select a term is dependent on the return data type defined for that term and the context of the selected trigger point.

Each operator defined in the Register Operators page supports certain data types; this determines which operators are available when you select a term in the condition builder. Furthermore, the selection of an operator determines how many fields are required on the right-hand side for entering values.

For example, if you select the is between operator, two right-hand side fields appear for entering values.

See Chapter 13, "Administering the Framework," Registering Operators and Operator Sets, page 158.

## Entering Right-Hand Side Values of a Condition

Prompt options that you specify when defining a term determine the right-hand side field type. The field may be a prompt, dropdown list, or multi-select prompt. SetID-based prompt tables specified in the term definition use the setID value specified on the policy definition page to perform a setID to setID indirection, thus constraining the prompt list.

---

**Note.** Use a semi-colon as a separator in multi-select prompts.

---

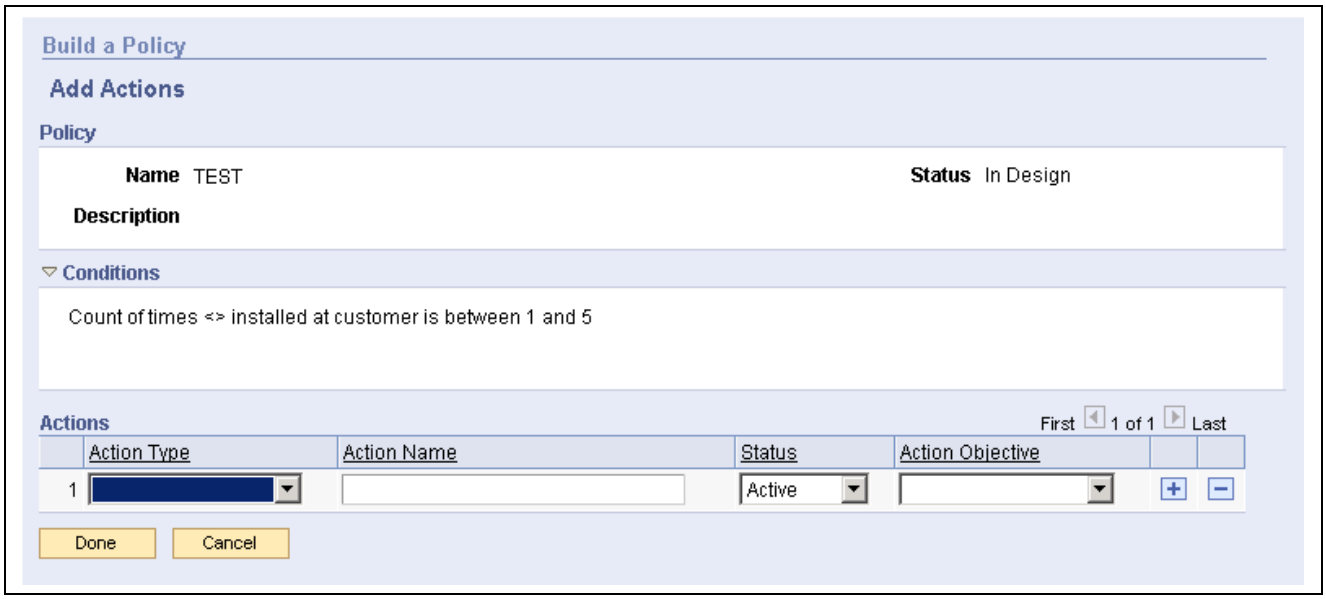
In advanced mode, you can specify a term on the right-hand side of a condition—this term is resolved at runtime and the resolved value used as the right-hand side value.

After you enter the condition, click Done to save the condition. Before a successful save, the system validates that right-hand side values are present and are of the appropriate data types; that parenthesis match; and that configurable terms have been configured.

## Adding, Editing and Configuring Actions

Click Add Actions or Edit Actions to display the Add or Edit Actions page.

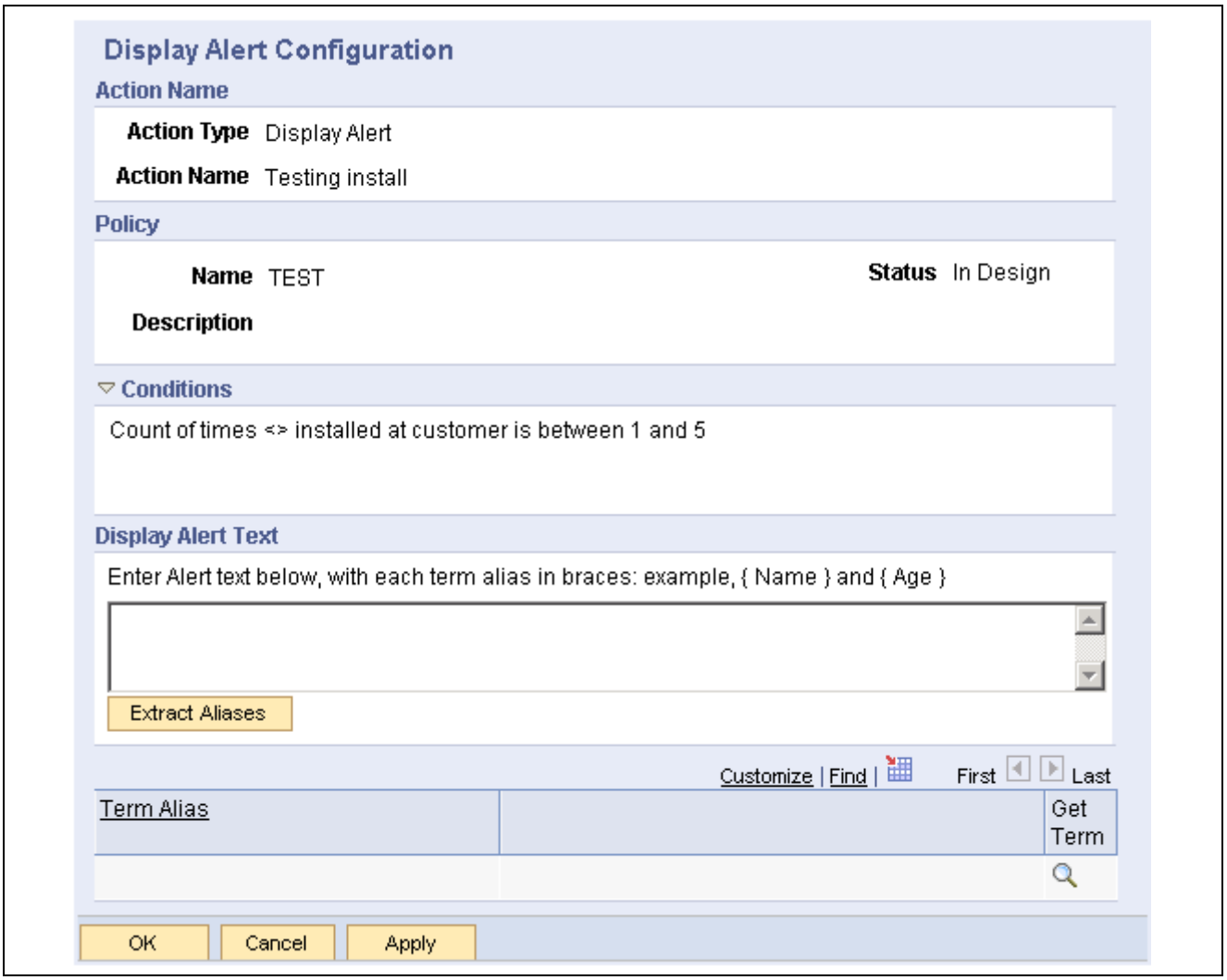
Copyright © 1992-2006, Oracle. All rights reserved.




Add Actions page

If the action type is configurable, the Configure button is enabled on the page. Individual action types have specific configuration pages.


Copyright © 1992-2006, Oracle. All rights reserved.




Configuration page for Action

**Note.** PeopleSoft Active Analytics Framework delivers a display alert action type. Other action types may be delivered by individual product lines. Please refer to the appropriate product line PeopleBooks to get more information on delivered action types.

See Chapter 12, "Setting Up the Action Framework," Registering Action Types and Action Type Bundles, page 145.

## Configuring the Display Alert Action

Access the Display Alert page by selecting *Display Alert* action type.

1. Click Configure.

   The Display Alert Configuration page appears.

2. Enter text in the Display Alert Text field and include any term aliases in braces.

   Term aliases are placeholders for dynamic content to be merged in the alert text at runtime.

3. Click Extract Aliases to extract term aliases to populate the grid, thus enabling you to map each alias to a term.

Copyright © 1992-2006, Oracle. All rights reserved. 

4. Click Get Term to map a term for each term alias in the grid.

   Only terms that return a single value can be used within the display text. Return data types of record or rowset, or terms with *Many* rows specified are not allowed.

5. Click OK or Apply to save the display alert configuration.

   This configuration is retrieved at runtime to generate the alert text and display it in a popup box.

### Activate a Policy

Access the Build Policy page.

Click Activate. The system sets the status to active after executing validations. Activating a policy disables field editing, however, the policy may still be copied and associated to another trigger point.

**Note.** On activation of the policy, any modifications made are in effect only in new user sessions. Therefore, you must sign off and sign on to see any changes made to the policy.

### Associating a Policy to Another Trigger Point

A policy can be associated to more than one trigger point within the same setID. Do this by:

- Adding a new row to the Associated Trigger Points grid.
- Selecting a valid trigger point from the dropdown list

**Note.** The trigger points available for selection in the drop-down list are constrained by the terms used in the policy condition and the actions configured in the policy. Also, a policy cannot be associated with multiple trigger points spanning multiple setIDs.

- Saving the policy associates the policy to the new trigger point.

### Copying a Policy

Create a new policy by copying an existing one, provided you can use the same condition and actions. While copying a policy, you'll be prompted for a trigger point and setID. Selecting from the list of valid trigger points results in creating a new policy, which appears on the screen.

**Note.** When you copy a policy, the condition and actions, but not the action configurations, are copied to the new policy. Therefore, you need to reconfigure the actions by clicking Edit Actions. A reminder message appears when you're transferred to the new policy.

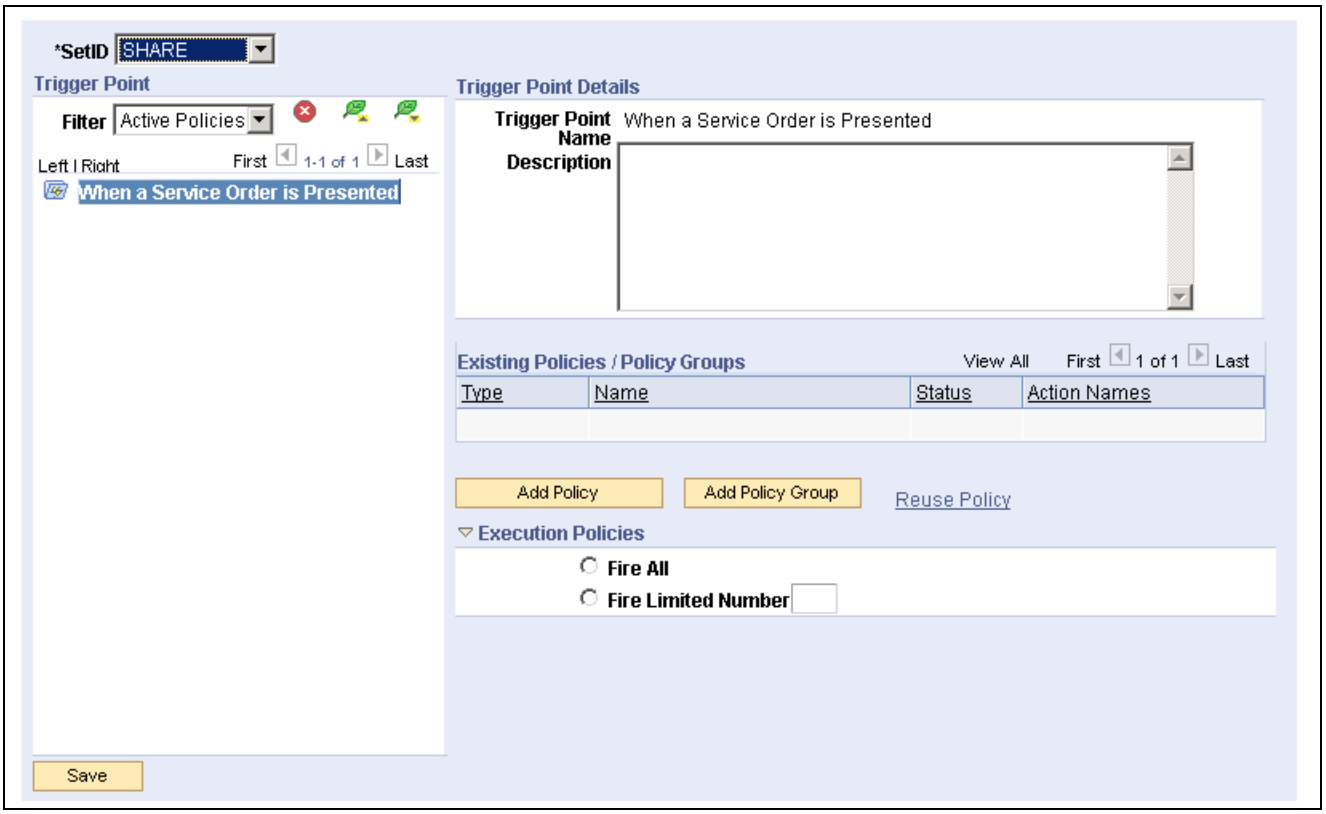
# Managing Trigger Points

The Manage Trigger Point component provides a comprehensive view of policies that are associated to a specific trigger point. It displays policies and policy groups in a hierarchy, with the trigger point as the root and policy groups (if any) as parents of policies.

In addition, this component enables you to assign execution options at the trigger point level and at the policy group level, facilitating policy arbitration and better policy management.

Access the Manage Trigger Point page.

 Copyright © 1992-2006, Oracle. All rights reserved.




Manage Trigger Point page

The trigger point hierarchy on the left-hand side of the page displays all the policies and policy groups associated to the selected trigger point. The trigger point appears as the highest level item in the hierarchy while policy actions appear as the lowest level items.

**Filter** This field applies only to policies displayed in the hierarchy and the Existing Policies/ Policy Groups grid. It either displays active policies, in-design policies, or all policies depending on the selection.

**SetID** Toggle the value in this field to view policies for this setID.

## Removing a Policy or Policy Groups

You can remove a policy or policy group from a trigger point by selecting the policy or policy group to be deleted, and then by clicking the Delete icon. Removing a policy or policy group from the trigger point disassociates it from the trigger point, but does not delete it from the database.

A policy group is not reusable—once the policy group is disassociated from a trigger point, it can not be referenced.

Any changes made to a trigger point by adding or removing policies or policy groups or by modifying execution options, take effect at runtime and only for new user sessions. Therefore, you must sign off and sign on to see any changes made.

---

**Note.** A policy that is associated to a single trigger point (either directly or within policy groups) cannot be removed from the trigger point. To disable such a policy, you must edit it and set the status to In Design.

---

Copyright © 1992-2006, Oracle. All rights reserved.

# Reordering Policy or Policy Groups

To set priorities to policies, you may need to reorder policies or policy groups within a trigger point or a parent policy group. Reorder policies and policy groups by using the reorder icon. a parent group, which may be a Trigger Point or a parent Policy group.

# Adding a Policy or Policy Group

Click Add Policy to create a new policy and associate it to the trigger point. The Build Policy component appears.

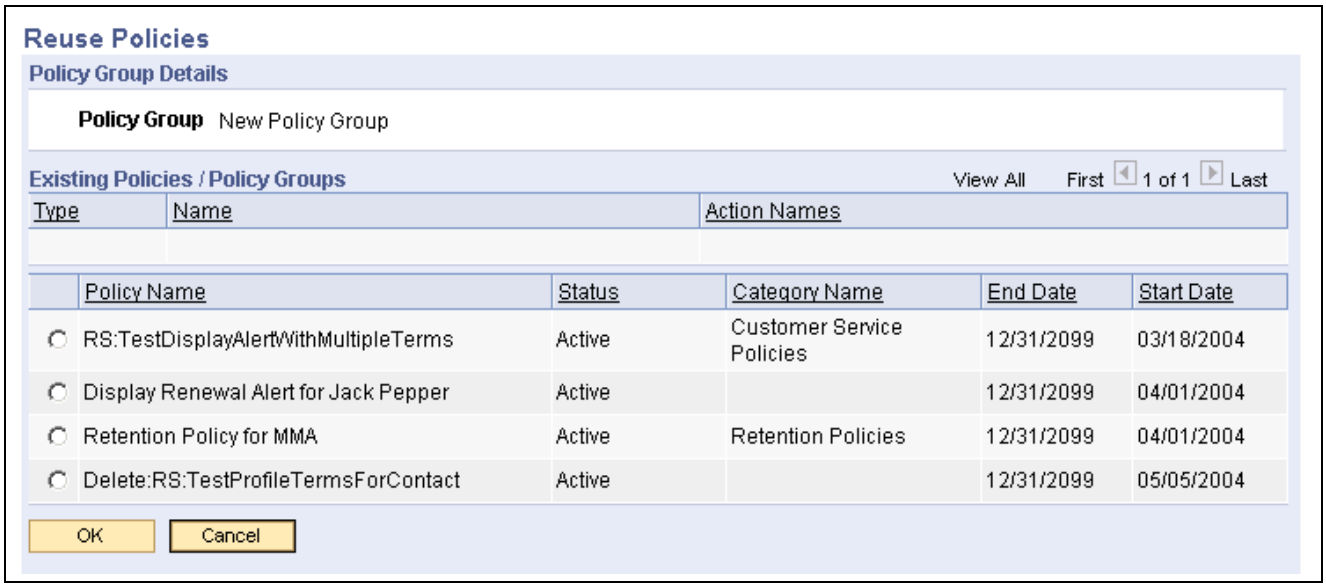
Click Add Policy Group to create a new policy group and to associate it with this trigger point.

A policy group may be used to set policy priority within a group, to de-activate policies, to nest child policy groups and to assign preconditions.

# Reusing Policies

Reuse a policy in multiple trigger points and policy groups if the contexts are compatible.

Click the Reuse Policy hyperlink to reuse an existing policy. Select one of the policies listed in the grid by selecting the appropriate radio button. Click OK. The selected policy is associated to the trigger point or policy group.




Reuse Policy page

**Note.** If a policy is reused within a single trigger point through direct association to the trigger point, or by association to a policy group within the trigger point, you cannot remove this policy from either the trigger point or the policy group within the trigger point. If you want to remove such a policy, you must deactivate the policy by setting its status to In Design.

# Adding a Precondition

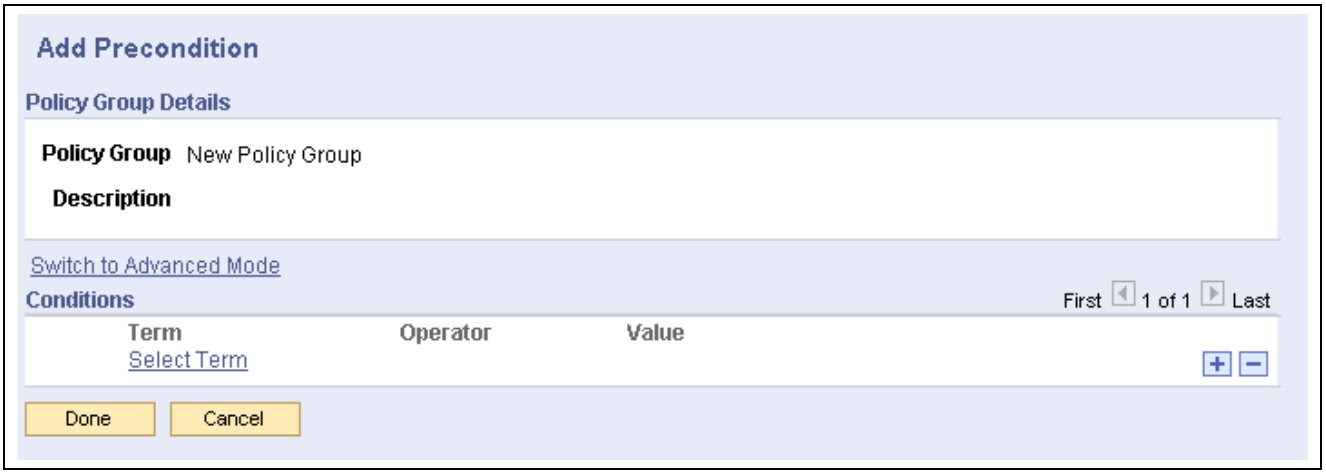
Preconditions are combinations of one or more conditions. They are optional and only policy groups can have preconditions.

At runtime, the policies within a policy group are not evaluated unless the precondition evaluates to true.


Copyright © 1992-2006, Oracle. All rights reserved.

For example, you could have a precondition defined for a self-service policy group, as "Is this user on the internet?" Consequently, all policies within that policy group would not be executed unless the precondition of being on the internet evaluates to true.

Click Add Precondition to display the Add Precondition page.




Add Precondition page

Select terms and operators; specify values on the right-hand side.

# Setting Execution Options

The following execution options may be specified for a trigger point or a policy group:

| | |
|---|---|
| **Execute All** | Default, if specified for a trigger point. Enables execution of all policies within a trigger point or policy group. During runtime, when executing policies within a policy group, the system adheres to the execution option specified for the policy group. |
| **Execute Limited Number** | Enables execution of a maximum (of the number specified) policies that evaluate to true. |
| **Use Parent Execution Options** | (Policy groups only). Default for policy groups. At runtime, this uses the execution options of either the trigger point or the parent policy group. |

# Understanding Execution Options

This section describes the execution options and various scenarios of how they can be used.

## Execute All

*Execute All* tests all policies in a trigger point; that is, all policies in the trigger point can cause actions to occur if their conditions prove true.

For example if a trigger point has three policies and the execution option is set to Execute All, all three policies are evaluated.

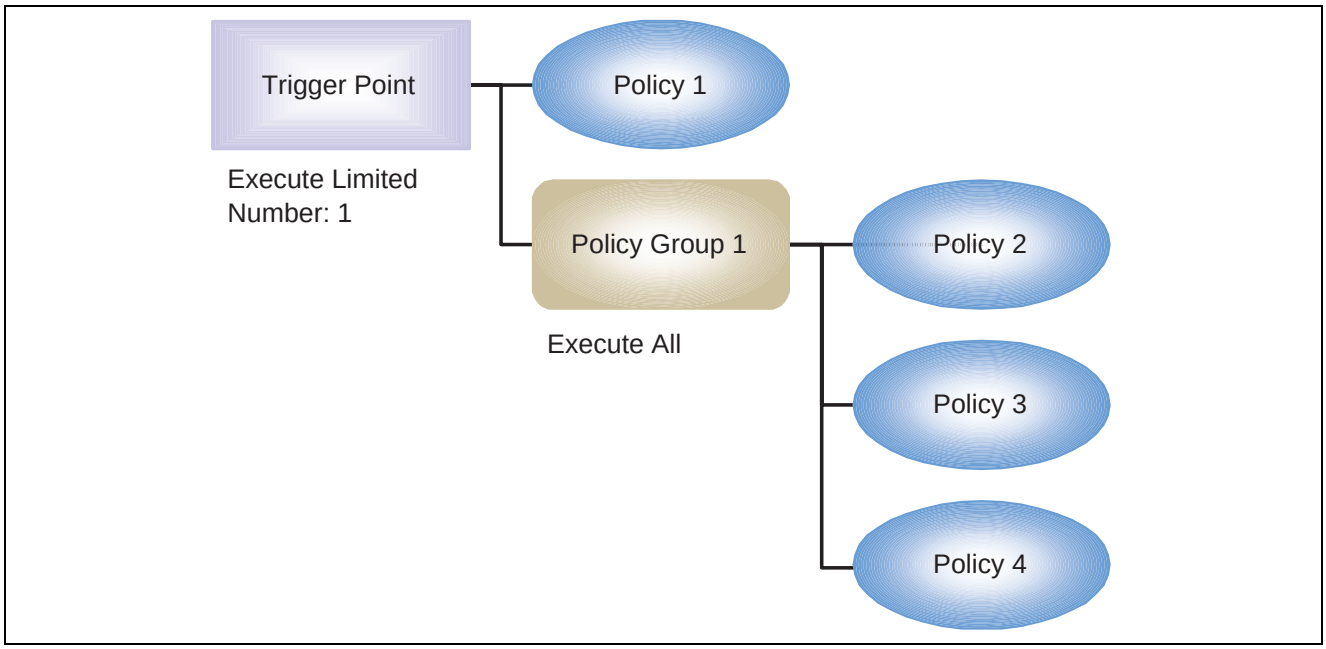
Copyright © 1992-2006, Oracle. All rights reserved. 

## Execute Limited Number

*Execute Limited Number* evaluates all policies in the trigger point until one of the policies' conditions evaluates to true. Therefore, say you set Execute Limited Number to 1 for a trigger point that has three policies where Policy 1 evaluates to false and Policy 2 evaluates to true; Policy 3 will not be considered and Policy 2 is executed.

## Various Scenarios of Execution Options

Policy groups may have their own execution options that could affect the option setting, however. For example, consider the following diagram:

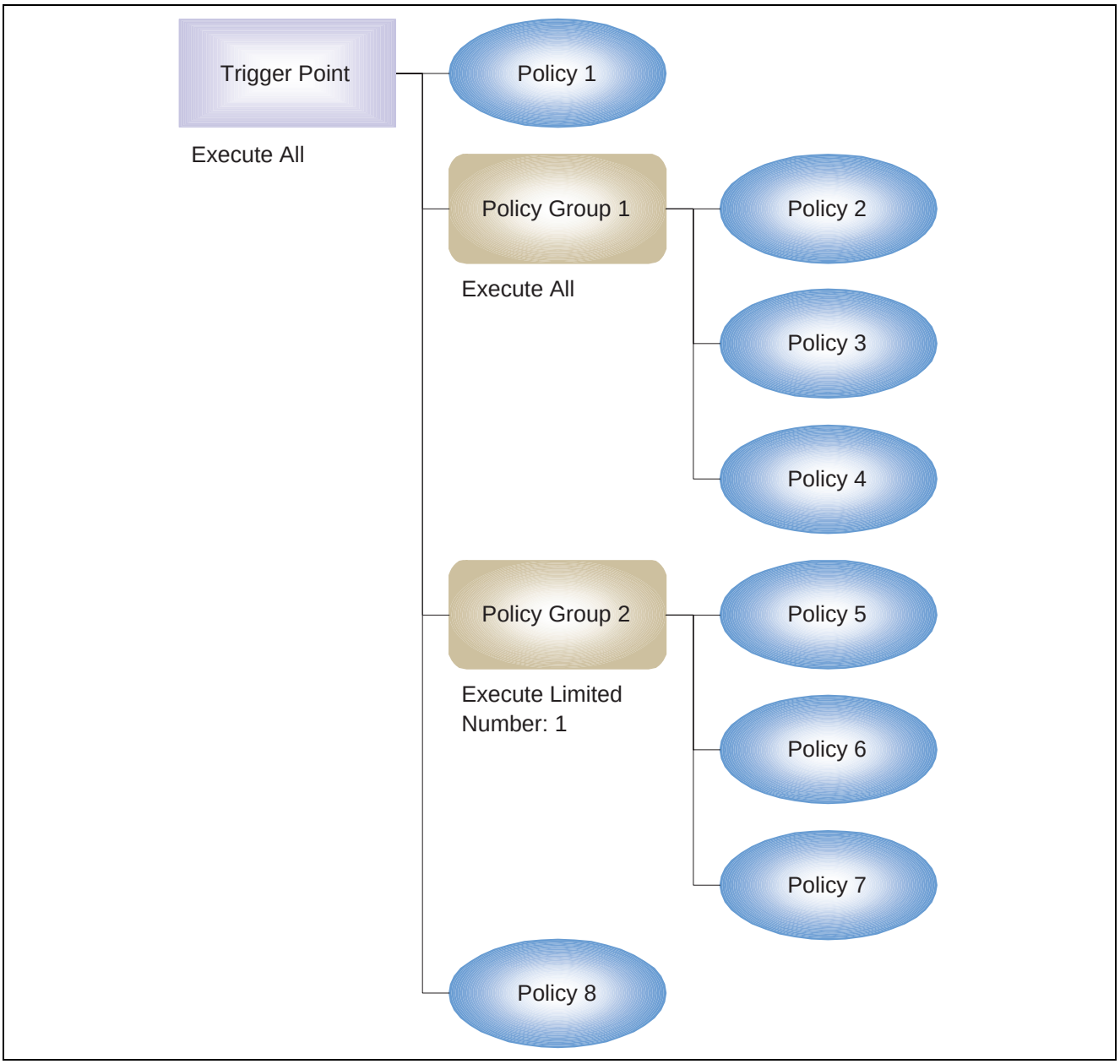



Execution Options: Scenario 1

In Scenario 1, if Policy 1 proves false, then all policies in Policy Group 1 are evaluated because its execution option overrides the trigger point's execution option. Therefore, even though the trigger point is set to execute one policy, if Policy 2, 3, and 4 evaluate to true those three policies' actions will execute.

Consider Scenario 2:


Copyright © 1992-2006, Oracle. All rights reserved.

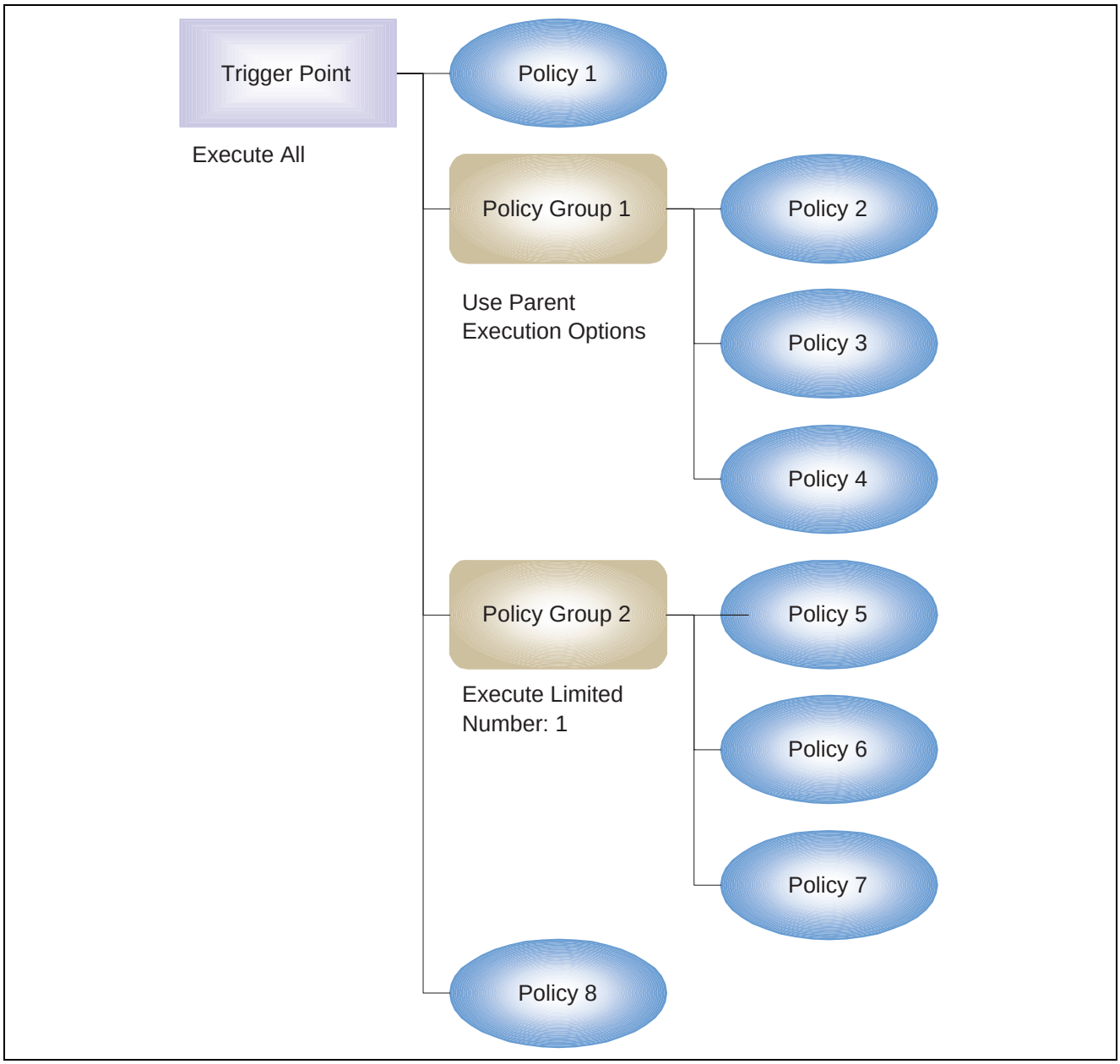



Execution Options: Scenario 2

In this scenario, assume that all policy conditions are true, actions from Policy 1, 2, 3, 4, 5, and 8 will execute.

Consider Scenario 3:

Copyright © 1992-2006, Oracle. All rights reserved.

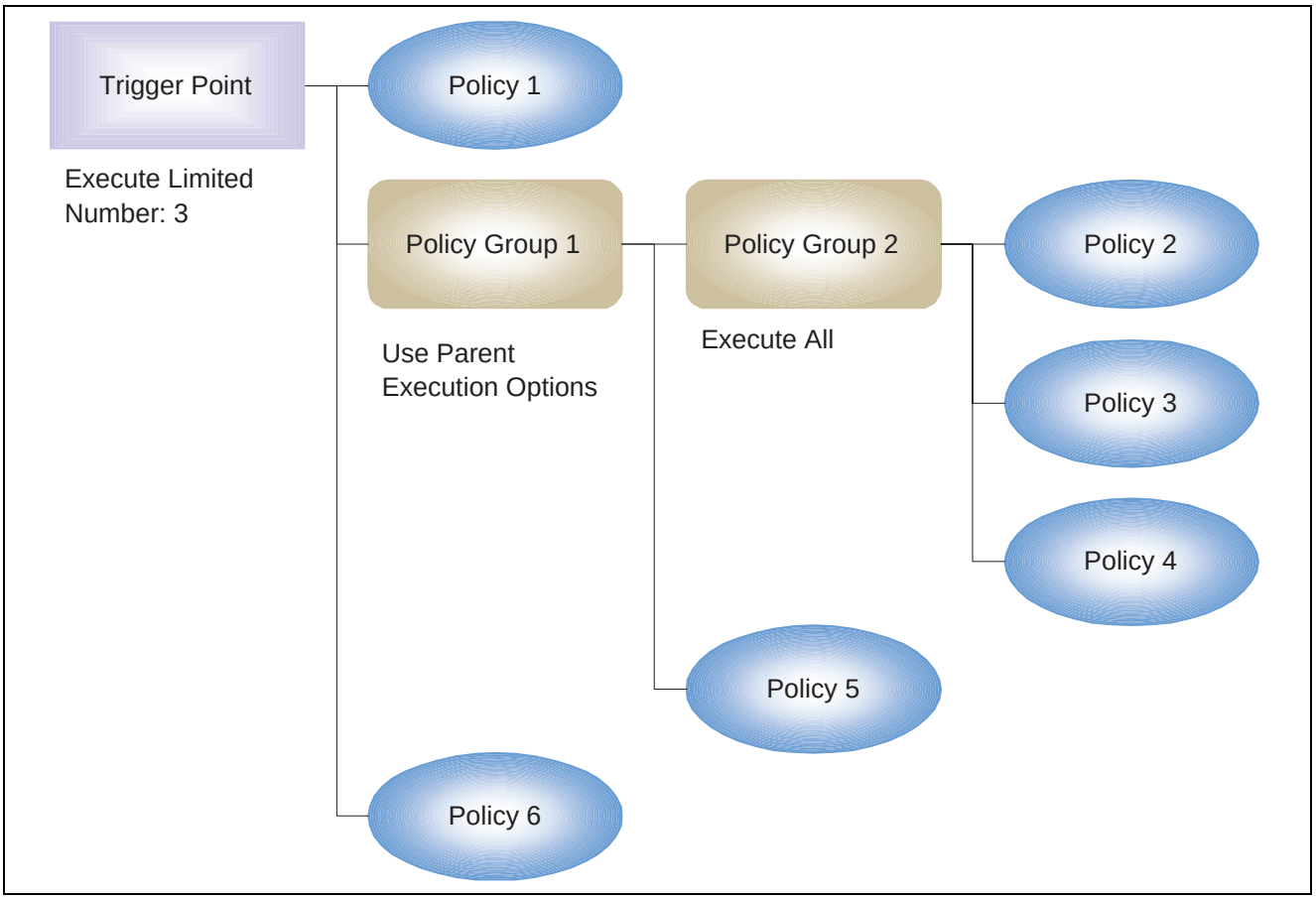



Execution Options: Scenario 3

In this scenario, the trigger point's execution option is the default (as is Policy Group 1). Assuming that all policy conditions are true, this trigger point executes exactly as in Scenario 2; that is, actions from Policy 1, 2, 3, 4, 5, and 8 will execute.

Consider Scenario 4:


Copyright © 1992-2006, Oracle. All rights reserved.




Execution Options: Scenario 4

In this scenario, assume that all policy conditions are true. Policy Group 2's execution option overrides Policy Group 1's; therefore, all of Policy Group 2's policy actions execute. However, Policy 5 is not considered, nor is Policy 6 because three policy actions have already executed (Policy 2, 3, and 4).

## Guidelines for Setting Execution Options

Unless you have a specific reason to set options, PeopleSoft recommends using the default execution options. If there are only a few conditions that could be true for a trigger point, and these conditions are logically exclusive (only one could be true at a time), set the number of policies considered to one (Execute Limited Number =1) to improve performance.

If for whatever reason, only a few policy options should be considered, and there are no policy groups, setting a limitation for the trigger point is a good idea.

If you have a trigger point containing unrelated policies and a category of possible conditions that may be true, for which only one set of consequent actions should be taken, it's best to separate the unrelated policies into a separate policy group with a limitation on the number of policies allowed to fire.

Copyright © 1992-2006, Oracle. All rights reserved.

 Copyright © 1992-2006, Oracle. All rights reserved.

# CHAPTER 10

# Setting Up the Data Library

This section gives an overview of the data library and discusses how to:

- Use the data library components.
- Create implementations.
- Create terms.
- Manage terms.
- Test term implementations.

## Understanding the Data Library

The data library is a repository for information within the PeopleSoft Active Analytics Framework. Each element in the data library is exposed by way of a *term*, which is a pointer to a unit of data within the PeopleSoft system. This data may reside in a relational database, or it may be derived at runtime.

### See Also

Chapter 8, "Understanding PeopleSoft Active Analytics Framework," page 107

## Using Data Library Components

This section describes the components used to populate the data library.

Copyright © 1992-2006, Oracle. All rights reserved.

## Pages Used in This Chapter

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Manage Terms | EOCF_TERM_CFGSRCH | Active Analytics Framework, Data Library, Manage Terms | Used to define and manage terms. |
| Subject Area | EOCF_TERM_SUBAREA | Active Analytics Framework, Data Library, Manage Terms, Subject Area | Used to define the subject area details. |
| Policy Options | EOCF_TERM_INACTION | Active Analytics Framework, Data Library, Manage Terms, Policy Options | Used to define policy options. |
| Extended Attributes | EOCF_TERM_ATTR | Active Analytics Framework, Data Library, Manage Terms, Extended Attributes | Used to add additional attributes to terms. |
| Define Implementation | EOCF_IMPL_DEFN | Active Analytics Framework, Data Library, Define Implementation | Used to define an implementation. |
| Test Term Implementation | EOCF_TEST_TERM | Active Analytics Framework, Data Library, Manage Terms, Test Term Implementation | Used to test a term's implementations. |

# Creating Implementations

Use the Define Implementations component (EOCF_DEFN_IMPL) to create and define implementations.

An *implementation* refers to the mechanism through which the data is retrieved, derived or computed. The implementation knows either where the data physically resides or it knows the algorithm for deriving the value. All terms must be associated with an implementation unless the data to which the term refers is present in the component buffer.

An implementation can be associated with more than one term. Conversely, a term may require multiple implementations if it needs to be resolved from multiple contexts. Typically, application developers or IT personnel develop implementations.

**Note.** Terms that are resolved by accessing data available in the current operating component's buffer, do not need implementations to be developed. PeopleSoft Active Analytics Framework provides mechanisms to access data available in the component buffer.

PeopleSoft recommends that when multiple related terms will be accessed during a single business event, create a single implementation to return a rowset containing the data for several terms; then, specify which data element or field position in the rowset or record is to be used for the term.

Implementations are developed using:

- Application class.


Copyright © 1992-2006, Oracle. All rights reserved.

Use application class implementations when retrieval or derivation of data involves writing procedural code; or, as a resolution method when data must be retrieved from an external source such as another PeopleSoft database or legacy systems. The application class can return data to the data library in a variety of forms: rowset, record, date, datetime, string, number, date array, datetime array, string array, number array and array of any. Use PeopleSoft Application Designer to develop the application classes.

---

**Note.** PeopleSoft does not recommend: 1) Using an application class to retrieve data from a component buffer; 2) Using an application class to retrieve the values for the binds directly by accessing the component buffer without registering them as implementation binds. Application classes must use APIs to retrieve the values for the implementation binds (input parameters).

---

- PS Query .

  Use Query Manager in PeopleSoft Application Designer to develop PS Query-based implementations. PS Query implementations are not appropriate for applications that get data from external databases or systems. The data library invokes the appropriate queries based on the information provided when you register the implementation. The data returned is available to the data library in the form of rowset.

- SQL object.

  Use this implementation when the SQL used needs to be platform independent and the data need not undergo complex transformations. SQL Object implementations are not appropriate for applications that get data from external databases or systems. The data library invokes the appropriate SQL object based on the information provided when you register the implementation. The data returned is available to the data library in the form of a record or an array of any objects. Use PeopleSoft Application Designer to create an SQL Object.

- Record.Field

  Create a Record.Field–based implementation when the data can be retrieved directly from a table without going through complex transformations. The data returned is available to the data library in the form of string, date, datetime, number, string array, date array, datetime array and number arrays.

## Registering an Implementation

With the exception of component buffer implementations, all implementations must be registered in the PeopleSoft Active Analytics Framework. Before registering an implementation, you should have already defined PeopleSoft Application Designer objects if using application class, PS Query or SQL Object implementation methods.

Specify the following in the registration component:

- Functional name.
- Resolution method used for that implementation
- Values for the parameters needed for invoking the implementation. The list of parameters vary depending upon the resolution method.
- List of binds that are expected by this implementation.

Copyright © 1992-2006, Oracle. All rights reserved. 

**Note.** The binds specified for an application class implementation are referenced in the application class object for retrieving the values. Therefore, changing these implementation bind names can have adverse effect on the term resolution.

*For IT users*: The list of implementation binds specified are being used for two purposes: 1) To allow implementations to access these bind values. For any implementation, bind values are passed by position regardless of the resolution method used. Application class-based implementations alone have the additional capability to access the bind values by name. 2) To allow the data library engine to use these binds to uniquely tag the data in application cache. If IT users "take a short cut" by retrieving the necessary data by directly accessing the context (by not registering the data as implementation binds), the data library engine may, as a result, tag the data with incomplete key information. This could cause the same cached data to be incorrectly reused for resolving terms for which it is not valid.

# Creating Terms

A term is a user-friendly name that refers to the data library content. It's essentially a piece of information that could exist in the PeopleSoft system or an external system, or it could be derived. For example, the data could be available in the component buffer; retrieved using a PS Query or an SQL object; or, computed using an application class.

Terms are the building blocks in policies. Functional users can build conditions for a policy using terms present in the data library. Terms must be registered in the PeopleSoft Active Analytics Framework before they can be used.

Registering a term is a multi-step process that includes:

- Developing an implementation.
- Registering the implementation.
- Defining the term.
- Associating the term to one or more subject areas.
- Testing and activating the term.

## Understanding Term Properties

Defining a term involves specifying the following:

- Term name, code and type (constant or variable).
- Data type.

  The data library supports primitive data types of string, number, datetime, date , time; and PeopleSoft-specific data types of record and rowset.
- Number of rows to be returned, scalar or vector (returning an array). (Terms that are record or rowset data types have number of rows set to *One*).

  **Note.** Terms returning a vector (where value of number of rows is many) do not appear in the term list while building a condition for a policy.

- User binds.

 Copyright © 1992-2006, Oracle. All rights reserved.

These are values that would be supplied either during the construction of a condition or at the time of associating the term with the application. Not all terms will have the binds; however, user binds may make a term more reusable.

- Optionally, details about how the data needs to be captured for user binds:whether a prompt or dropdown list needs to be shown and how to derive the values.
- Which implementation needs to be used for resolving the term.
- Whether the term can potentially be resolved from any context; or only from specific contexts
- How the data library needs to extract the data from the content returned by the implementation.
- Prompt details for the term.

  Specifying prompt details for a term is needed only when the term will be used to build a condition. The prompt details convey how the data needs to be captured on the right-hand side for a term participating in a condition.
- Configuration details for prompts.

  The details you provide are used during the construction of a condition. When a term is selected as an element in a condition, the right-hand side widget will be constructed based on the configuration details specified for the prompts. You can configure the following prompt types :

  - Translate.

    Specify a translate field name in which its values appear to the end user on the right-hand side of a condition.

  - Dropdown.

    Specify a record name, data field name, and description field name. The record and data field names supply the valid data values to display in the dropdown list; the description field provides a user-friendly description of the data value.

  - Prompt.

    Specify a record name, data field name, and description field name. The record and data field names supply the valid data values to display in the prompt; the description field provide a user-friendly description of the data value.

  - Custom.

    Specify a custom application class, a data field name, and a description field name. Valid values are retrieved by executing the specified application class method and presented as a prompt.
- Scope of each term implementation.
  - When caching is activated for a term, data that is cached is uniquely identified by the implementation ID and all of the implementation bind values for that implementation.
  - When scope is specified as the trigger point, after the first invocation of a term, subsequent references to the same term in one or more policies associated to the same trigger point forces the data library engine to retrieve the data from the cache, provided all the values for the implementation binds match those of the ones belonging to the data present in the cache.
  - When scope is defined as a component, the longevity of the data is for a specific instance of the transaction.
  - When scope is defined as global, the cached data is available for the entire user session.
  - When scope is defined as *Do Not Cache*, data is retrieved by invoking the implementation every time.
- Association of a term to subject areas.

Copyright © 1992-2006, Oracle. All rights reserved. 

Subject areas act as file cabinets. You must assign a term to at least one subject area, but you can associate it to more than one.

**Note.** PeopleSoft Active Analytics Framework does not format the data. It is the term user's or term implementer's responsibility to format it according to their needs. For example, the term Current Date is always resolved using the standard YYYYMMDD format.

## Using Generic Implementations

A generic implementation can resolve terms within the requesting context. You define generic implementations for terms when they can be used in various contexts and when any new contexts may want to use that term.

Examples of generic implementations are:

- Customer-specific measures such as customer value, the number of cases reported in a period of time or the number of telephone interactions with the customer.
- Customer profile information such as, first and last name, email address, customers within a segment.

## Using Contextual Implementations

If the input data needed for invoking an implementation is too specific and cannot be supplied outside of the component, then the implementation must be associated with the component's context. For example, terms such as case status, order creation date, or case description, cannot be resolved from components other than those in which they are present.

Terms that have different implementations depending upon their contexts will have an implementation associated with a specific context. For example, the term *Revenue for a customer/ segment / segment group*is computed differently depending on the context from which it originates. The implementation specific to the customer context calculates the revenue value from that customer. The implementation specific to a segment context calculates the revenue value generated from all the customers belonging to that segment, and so on for segment group.

# Managing Terms

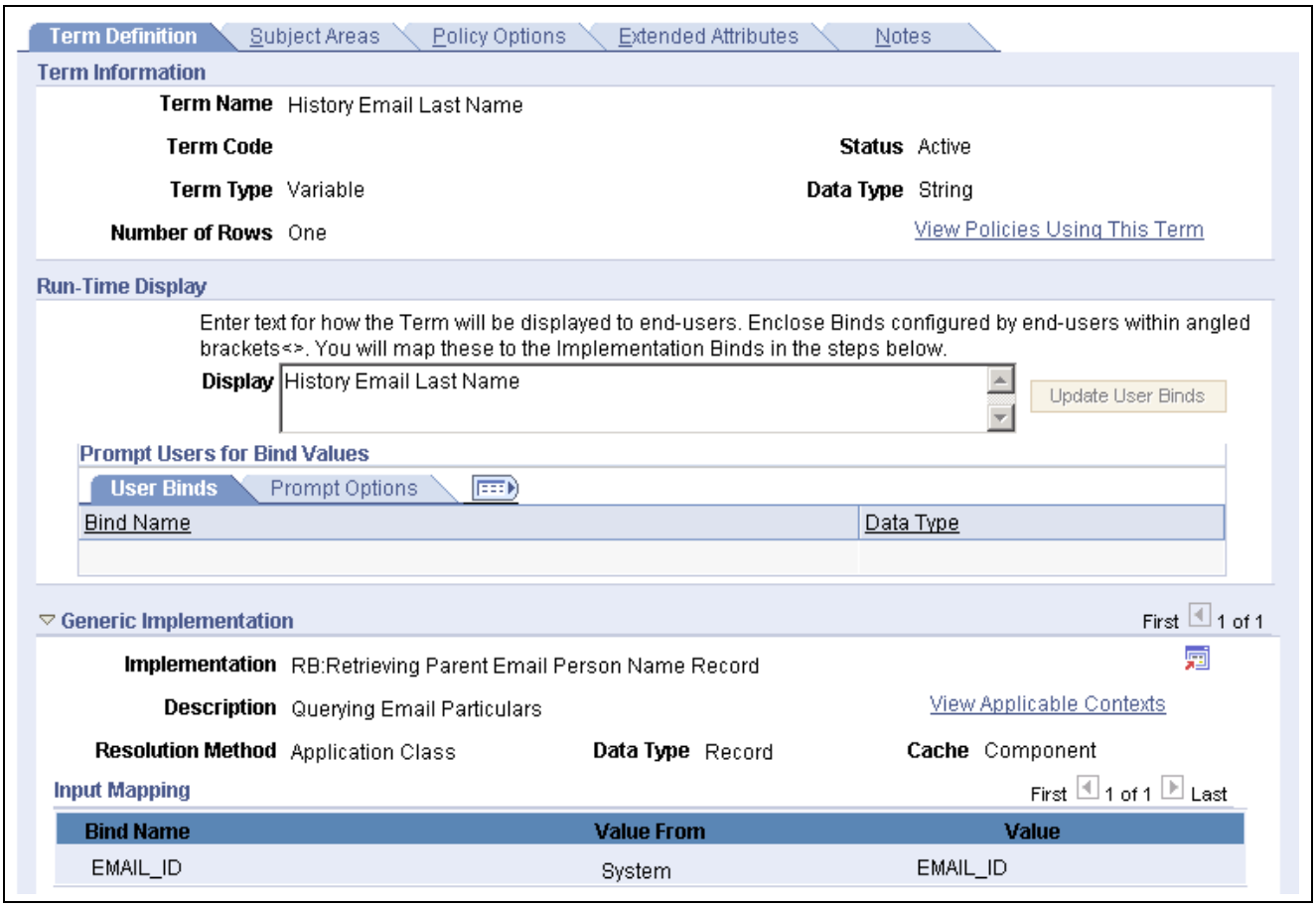
Before defining a term, you should have already:

- Created context definitions if you're using a contextual implementation.
- Registered any implementations if the term is not accessing the component buffer.
- Created prompt records in PeopleSoft Application Designer if prompt options need to be specified.

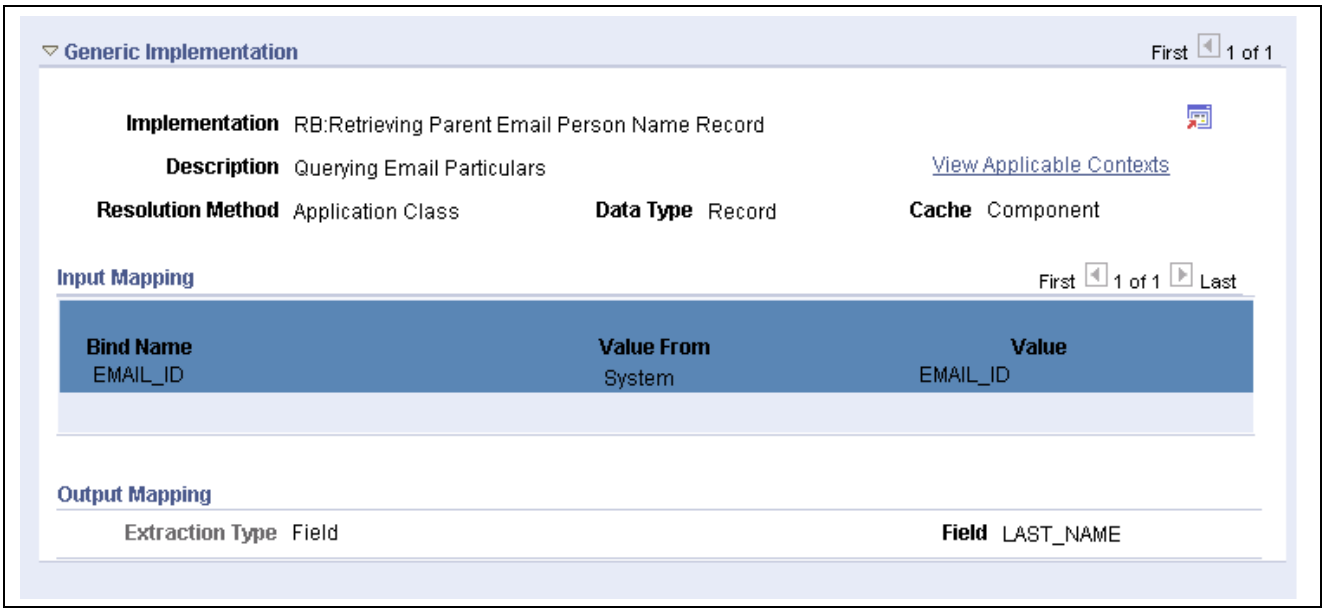
To manager terms, use the Manage Terms component (EOCF_TERM_DEFN).

Access the Manage Terms page.

 Copyright © 1992-2006, Oracle. All rights reserved.




Manage Terms page (1 of 2)




Manage Terms (2 of 2)

**Term Name** Unique identifier of the term; label that will be displayed to the functional users. Though allowed, PeopleSoft recommends that special characters not be used in term names.

Copyright © 1992-2006, Oracle. All rights reserved.

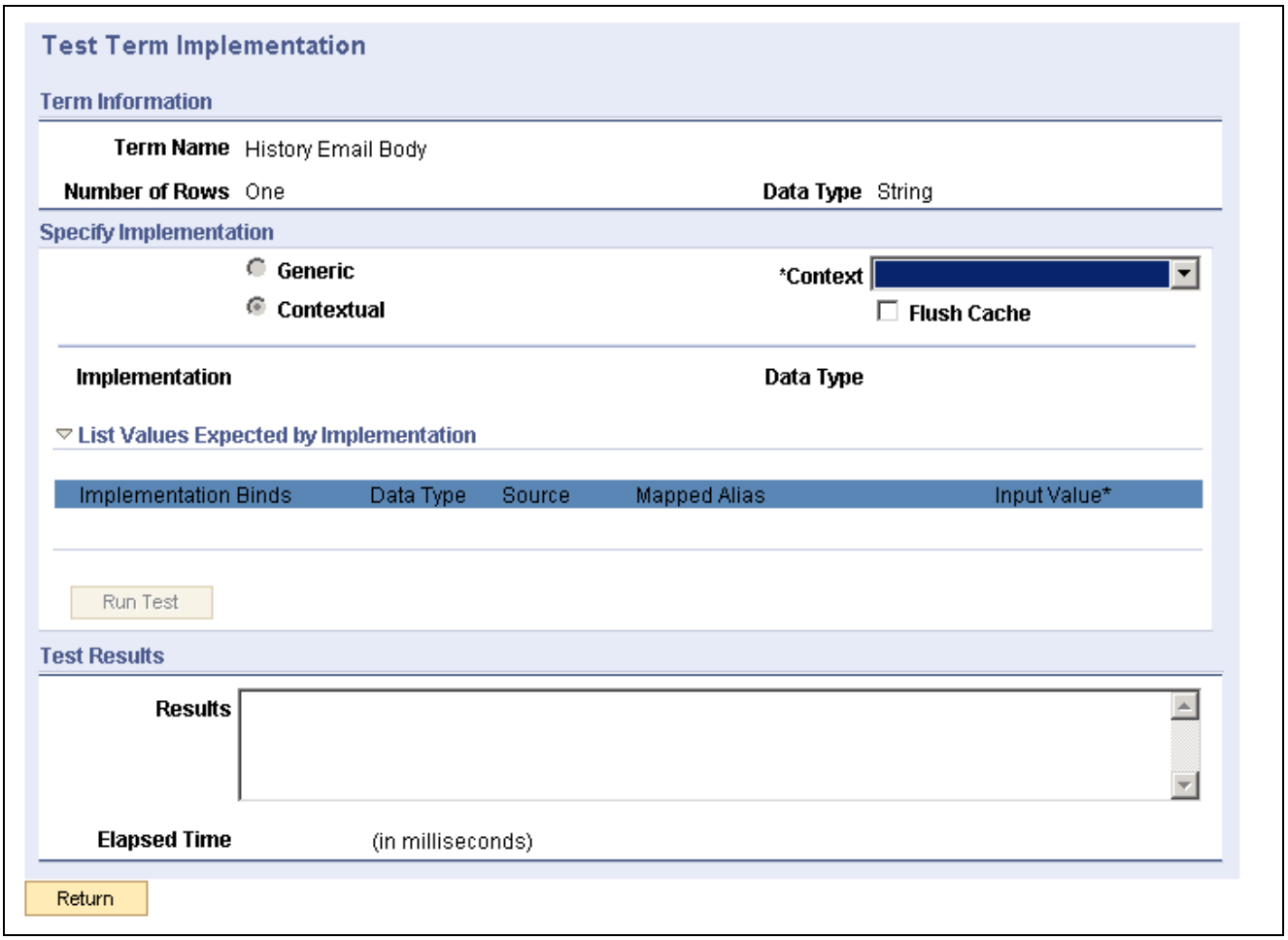
| | |
|---|---|
| **Term Type** | Specify that a term is a variable or constant. Variable terms must have at least one implementation. |
| **Term Code** | Uniquely identifies a term when accessing a term programmatically. This is user-defined. |
| **Number of Rows** | Number of rows to be returned, one or many (scalar or vector). If this field is *Many*, the term cannot participate in policy conditions. |
| | **Note.** Applications directly integrating with the data library are responsible for converting the resolved output value of a term (which will be of data type *any*) to the appropriate data type. |
| **Status** | Valid values are *Active* , *Inactive* and *In-Design*. Only active terms are used in policy conditions and other applications. |
| **Data Type** | Return data type of the term. Possible values are string, number, date, datetime, time, record and rowset. |
| **Run-Time Display** | Specify user binds for this term, which will be needed when the resolved value of the term depends on user defined binds. |
| **Prompt Users for Bind Values** | Specify the bind name; (optional) specify prompt options. |
| **Generic Implementation** | Specify a generic implementation. Generic implementations are resolved by deriving the bind values from the runtime context. Terms having generic implementations can be resolved by multiple contexts. You specify a generic implementation by selecting an existing implementation from the prompt or creating a new one using the Create button. Click on the View applicable contexts hyperlink to view the list of contexts in which this term would be resolved. |
| **Contextual Implementation** | Select an implementation specific to a context. Contextual implementations are resolved by deriving the bind values from this specific context. |
| **Input Mapping** | Maps the implementation binds to context variables or to constant values. If the term has user binds, one or more implementation binds must be mapped to the user binds. For generic implementations, this mapping is critical for this term to be resolved by multiple contexts. |
| **Output Mapping** | Specify the extraction parameters for a term implementation—such that a subset of the value returned by the implementation is returned as the resolved value of the term. |

**Note.** Use caution when making changes to the term definition after the term has been associated to one or more policies. Changes to term attributes such as data type, number of rows, implementation category, and implementation details; or, changing a non-configurable term to a configurable term and vice versa, could have significant impact on the policies which reference this term. It is possible that these changes could result in invalidating these policies. Before making any of these changes, view the policies using a term by clicking the hyperlink View Policies Using This Term.

# Testing Term Implementations

Access the Test Term Implementation page.


Copyright © 1992-2006, Oracle. All rights reserved.




Test Term Implementation page

**Specify Implementation** Specify whether you want to test the generic or contextual implementation defined for the term.

- Contextual. If you select this you must specify a context name from the drop-down list. Displays the list of bind values required by this implementation in a grid.
- Generic. If you select this, the context from where the term is resolved must be specified.

Check Flush Cache if you do not want the system to fetch the value for this implementation from the memory cache.

**List values..** Enter sample values for the parameters expected by the implementation and click Run Test.

**Test Results** Displays the resolved value of a term implementation being tested and the elapsed time to retrieve the value.

---

**Note.** Context variable implementations of a term cannot be tested. Also, terms that have application class implementations accessing data from a component buffer or directly from the context, cannot be tested in the Term Tester page. Testing these terms will result in an error message.

---

Copyright © 1992-2006, Oracle. All rights reserved.

 Copyright © 1992-2006, Oracle. All rights reserved.

# CHAPTER 11

# Managing Contexts

This chapter provides an overview of contexts and discusses how to configure contexts.

## Understanding Contexts

Contexts are a key component of how the PeopleSoft Active Analytics Framework works—their purpose is to describe the computing environment from which the decision engine is invoked and select the appropriate term implementation at runtime.

### Online and Generic Contexts

Contexts can be online or generic. Online contexts must be associated with a PeopleSoft online page, whereas generic contexts can be used anywhere (including online pages).

The data elements within a context are called *context variables*. Online contexts have built-in context variables—they are the fields of the online page. However, the data elements in an online context are not limited to the online fields; both online and stand-alone contexts can have additional context variables defined.

Within a given context, all context variables must be assigned unique code names called *aliases*. Aliases are used to associate data with the input binds required by term implementations. For example, a field representing the customer ID may be named CUSTOMER_ID in the application page, but may have an alias of CUST_ID. Using aliases enables terms to be used in the largest possible set of contexts. Page variables exist on a one-to-one basis with their underlying fields; that is, two or more context variables may point to the same page field as long as the aliases of these context variables are distinct. This enables the context user to have aliases named both CUSTOMER_ID and CUST_ID, and thereby facilitate term re-use. Context variables are exposed to the user by corresponding term definitions.

At runtime, applications may request the data library engine to automatically populate the online context in memory, provided the request is made from the component from which the online context can be constructed. In case of generic contexts, applications can either construct the context explicitly by populating values for these context variables, or request the data library to automatically copy values from level 0 context variables of an online context to similar context variables of a generic context.

Context variables can be:

- Page variables that correspond to fields within PeopleSoft application pages.

  Page variables are also referred to as native context variables, as they are native to the pages from which their values come.

  **Note.** Use caution when altering the component structure after generating page variables for an online context (using Generate Context component). Changes done to existing fields on the application page might invalidate the corresponding context variables and the terms from which they're resolved.

Copyright © 1992-2006, Oracle. All rights reserved.

- Constants.

  Generic contexts usually consist of named constants. Constant-type context variables may not have specific values associated to them when the context is registered; some of these variables may get values at runtime.

- Term variables.

  These context variables are data library terms that have been inserted into the context. A context may have any number of terms included within it. However, if a term's implementations require binds, then the context must provide them from its page variables and constants. Term variables can be used to add extended data elements for implementation binds and actions without having to customize the application.

# Configuring Contexts

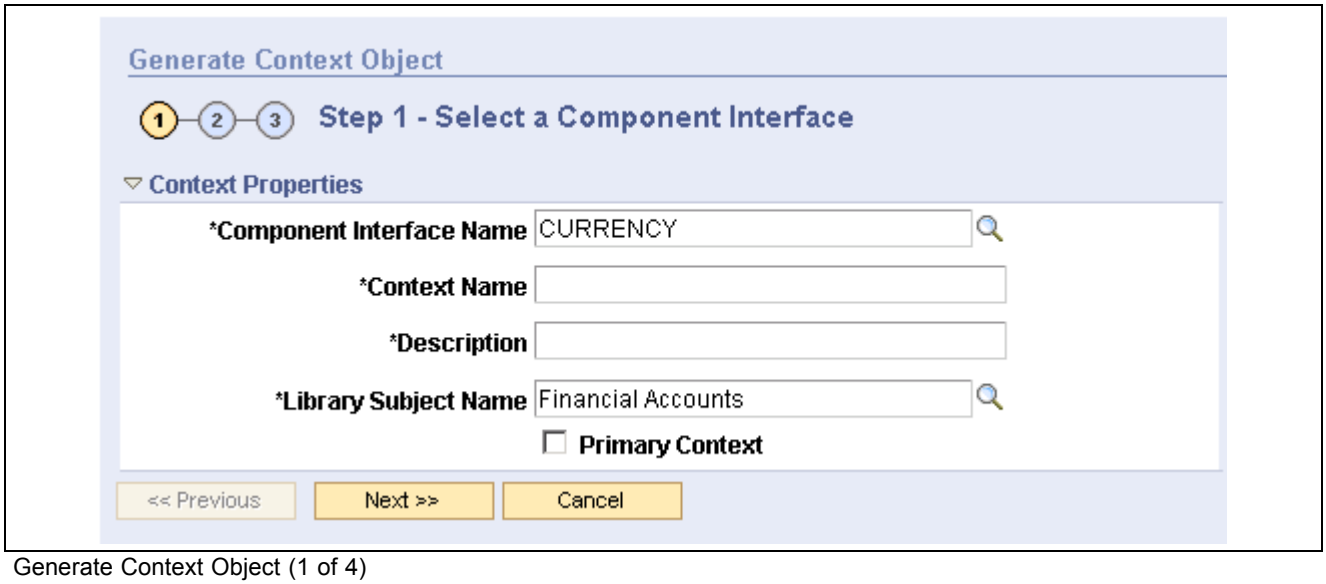
This section describes the components used to configure contexts.

## Pages Used in This Chapter

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Generate Context Object | EOCF_CTX_IMPORT | Active Analytics Framework, Setup, Generate Context Object | Used to generate context objects. |
| Manage Context Object | EOCF_CONTEXT_DEFN | Active Analytics Framework, Setup, Manage Context Object | Manages context objects. |

## Generating Context Objects

Access the Generate Context Objects page.




Generate Context Object (1 of 4)

**Component Interface Name** Used to extract the contents of a PeopleSoft component for use in generating the context. A component interface is required to create an online context.

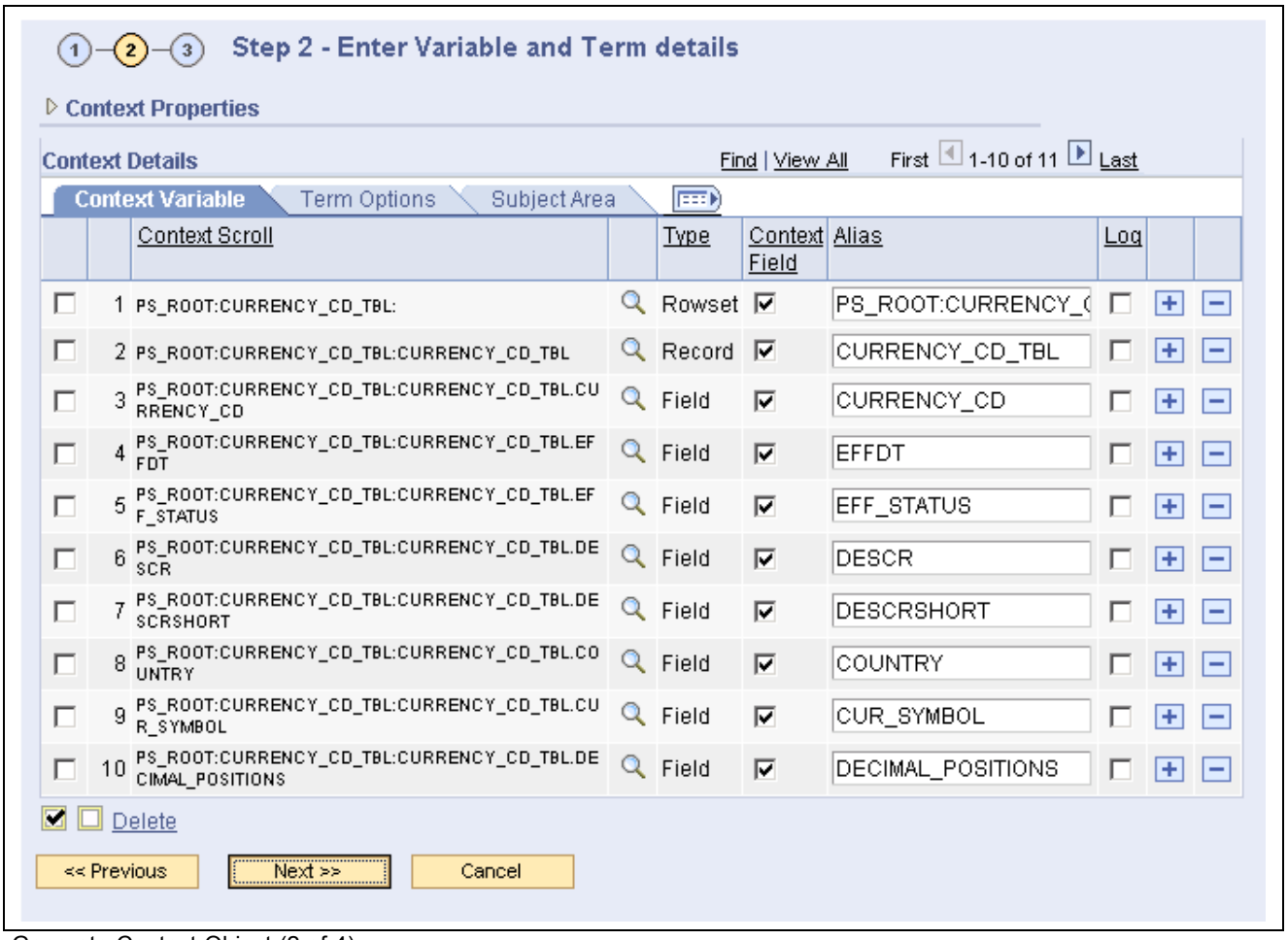

Copyright © 1992-2006, Oracle. All rights reserved.

**Context Name** Required, unique and descriptive name of a context.

**Description** Required, appropriate description about the intent of this context.

**Library Subject Name** Default subject area for terms created by this process. The prompt for this field shows the subject area selection list.

**Primary Context** Select this checkbox to denote the context to be generated as the primary context for this online transaction.



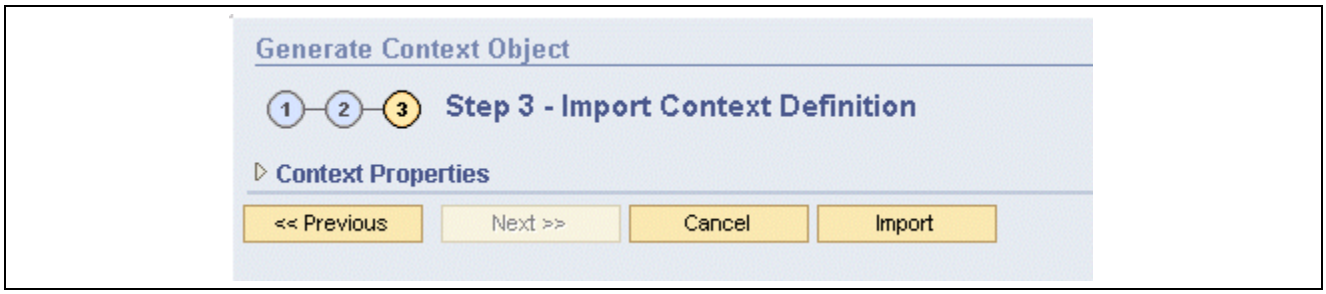

Generate Context Object (2 of 4)

Review and configure the context variables and terms that were automatically generated based upon the component interface you selected. Select the Log checkbox to denote that the field should be logged when a context is persisted at runtime (this is a key to the transaction, and the process automatically sets this field).

Select the Term Options tab to create a new term for each context variable that is created. Also, you can add a contextual implementation to an existing term for this context variable

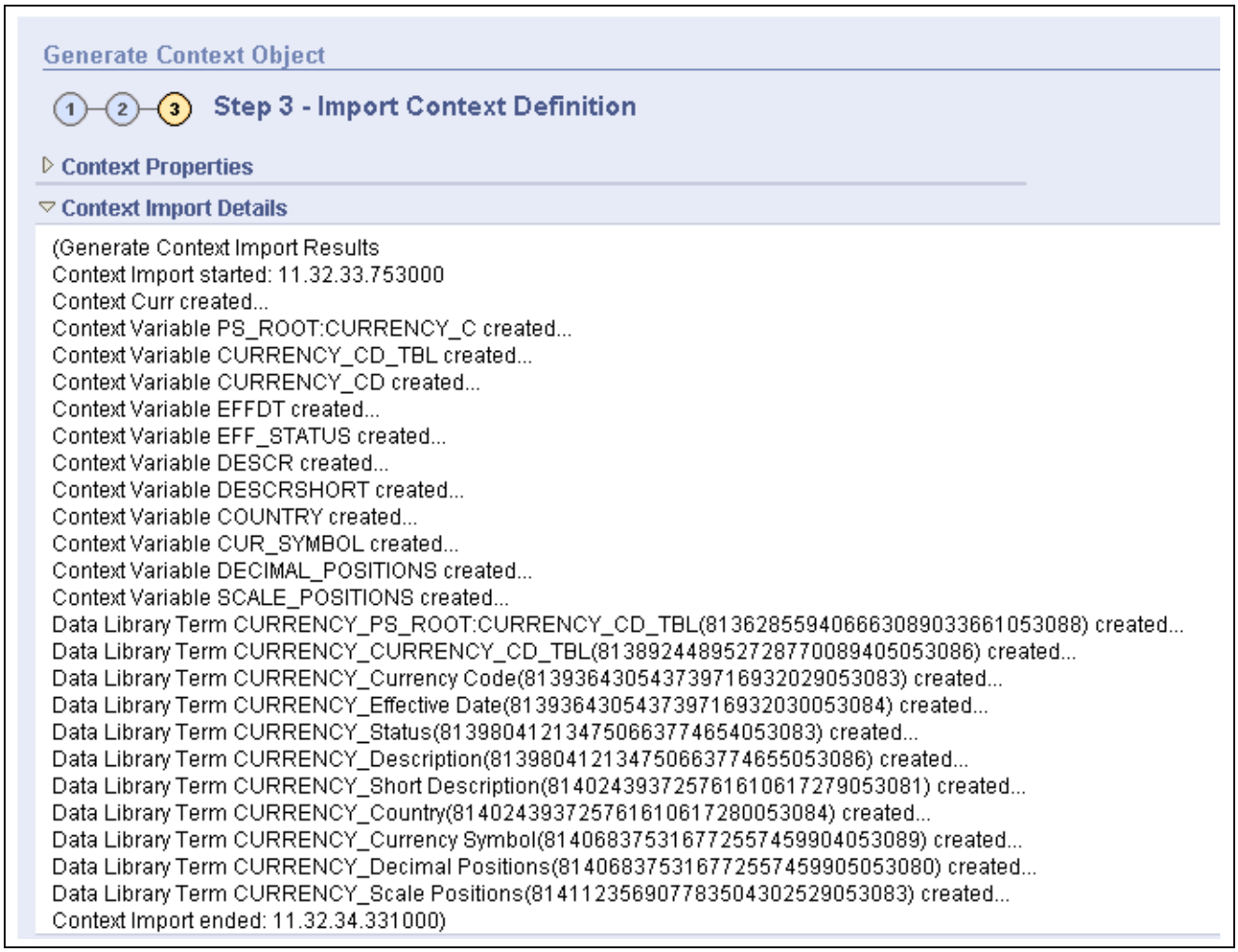
Select the Subject Area tab if you want to override the default subject area chosen on the first page.

Copyright © 1992-2006, Oracle. All rights reserved. 




Import Context Definition page (3 of 4)

Select Import to create the context and terms. The Generate Context Object screen appears listing the import details.

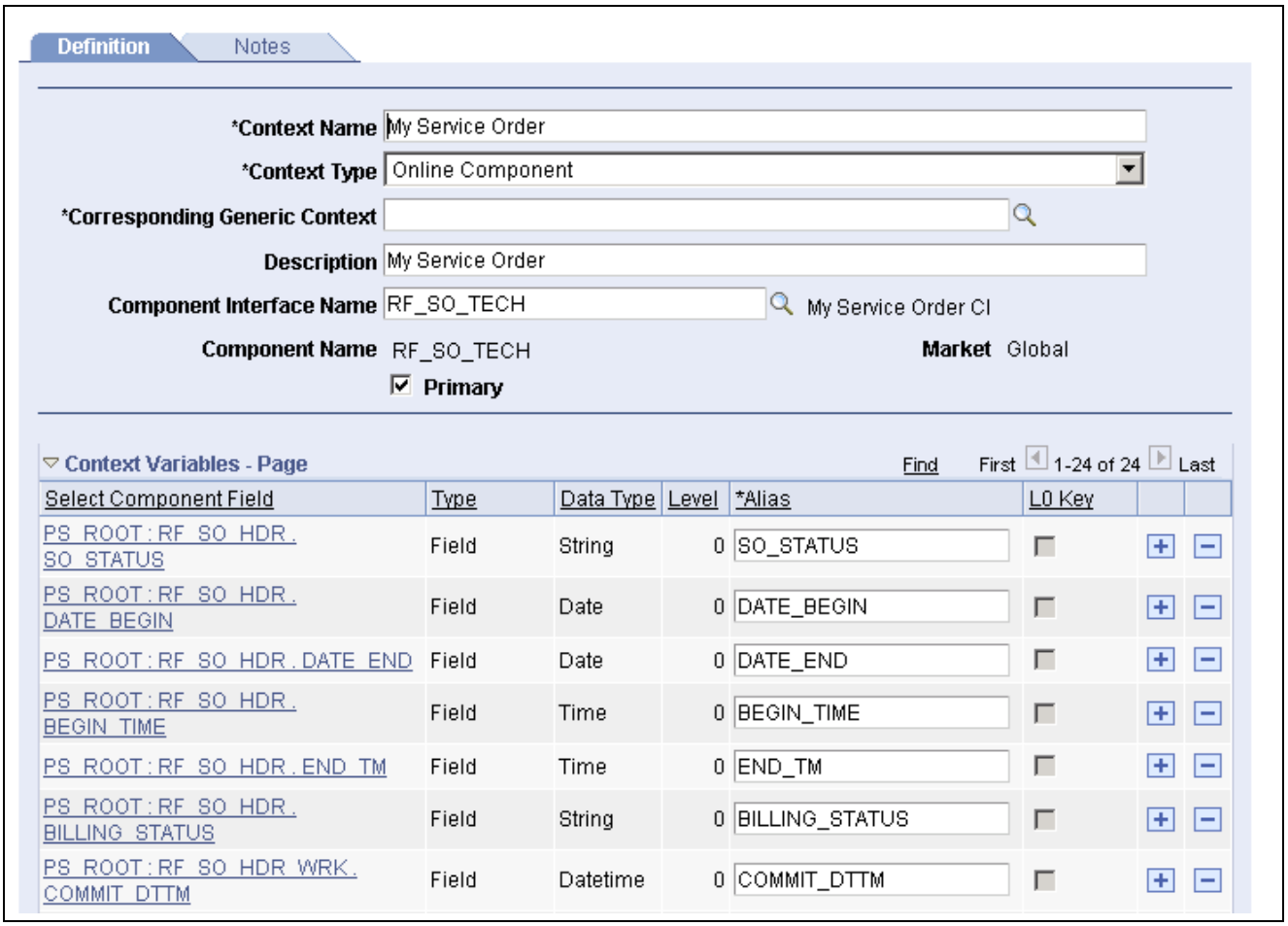



Generate Context Object (4 of 4)

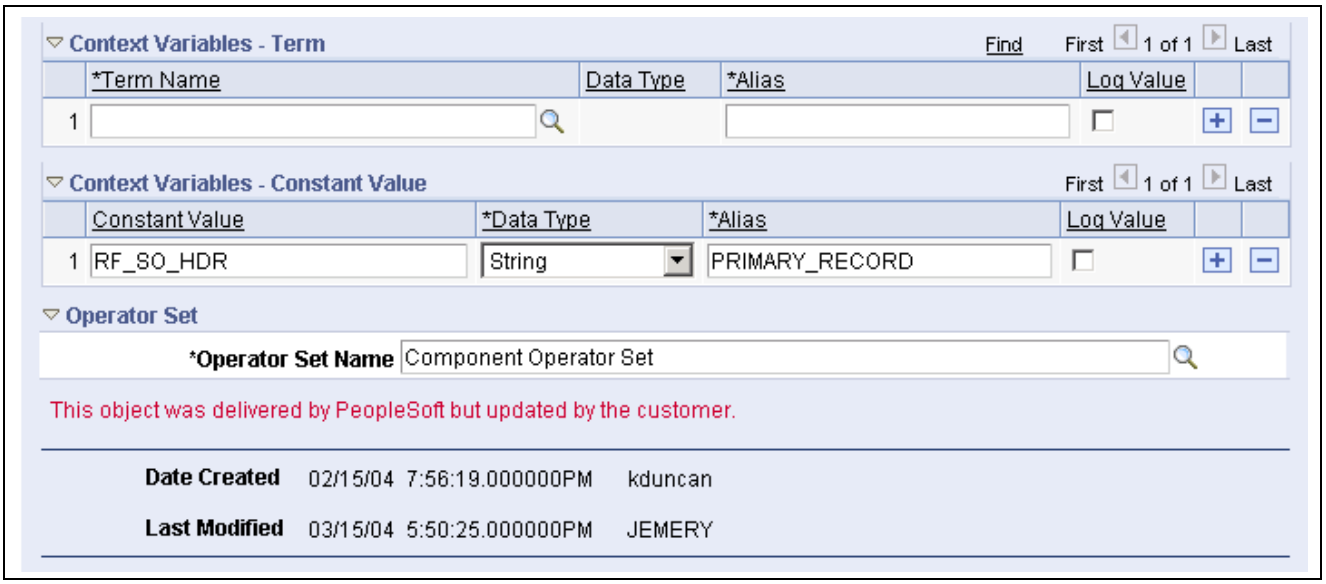
# Managing Context Objects

Access the Manage Context Object page.


Copyright © 1992-2006, Oracle. All rights reserved.




Manage Context Object (1 of 2)




Manage Context Object (2 of 2)

**Corresponding Generic Context** This field is used to programmatically create a standalone context from its corresponding online context. Applications directly embedding data library

Copyright © 1992-2006, Oracle. All rights reserved. 

terms need this information to supply context information to the data library runtime engine.

**Context Variables Page** This section is used to define the page context variables.

- Select Component Field. Click to select a component field from the displayed component buffer hierarchy.
- Type. This column specifies the object type of the selected component field. Valid values are *Field, Record, Rowset*.
- Data Type. Specifies the PeopleSoft data type of the selected component field.
- Level. Specifies the scroll level of the selected component field.
- Alias. Unique identifier for a context variable.
- L0 Key. Specifies whether a component field is a key field for this component.

**Context Variables Term** This section defines the context variables of type term.

- Term name. Select a term name from the prompt. The resolved value of this term is used as the value of this context variable.
- Data Type. Specifies the data type of the term selected.

**Context Variables Constant** This section defines the context variables of type constant.

- Constant Value.
  - Specify a constant value to be used as the runtime value for this context variable.
  - Leave blank and a value has to be supplied at runtime.
- Data Type. Specify the data type of this context variable.

---

**Note.** A constant value specified for a context variable cannot be overridden at run time.

---

**Operator Set** Optionally, specify an operator set to be used for this context. This operator set is used to present the list of valid operators for selected terms in policy conditions. A default operator set is automatically filled in.

**Notes Tab** Enter descriptive information in the text box.


Copyright © 1992-2006, Oracle. All rights reserved.

# CHAPTER 12

# Setting Up the Action Framework

This chapter provides an overview of the action framework and discusses how to:

- Register action types.
- Register action type bundles.

## Understanding the Action Framework

The action framework allows users to specify the actions to be executed when policy conditions evaluate to true within a trigger point.

### Architecture of the Action Framework

The architecture of the action framework comprises:

- An action type registration component for creating and maintaining action types by programmers and IT staff. An action type is metadata pertaining to a particular class of actions that might be performed at runtime.
- An action type bundle registration component for creating and maintaining action type bundles, or classes of actions that can be combined. An action type can be in only one action type bundle.

  **Note.** In this release, if other display action types are created by a PeopleSoft product line, they must be combinable with the delivered display alert action type, and must be registered in the display action type bundle.

- A design-time environment facilitating the embedding and configuring of consequent actions within policies.
- A runtime environment that allows the decision engine to invoke particular actions as needed.
- A generic display alert action type, which is available to all product lines using the framework. It can be used to display important information about a customer or a suggestion for a course of action. For example, if a high-value customer calls on the phone, a popup window appears with an alert and a suggestion that the phone connection be routed to a manager.

## Understanding Action Types

An action type refers to a category of actions that can be associated with a policy. For use in the framework, an action type must be registered with the following information:

- Location of the code that handles the design-time and runtime aspects of the action type.

Copyright © 1992-2006, Oracle. All rights reserved.

- Configuration requirements.

  When adding actions of specific action type to a policy, policy-specific configuration requirements must be set before the policy can be enacted.

- Whether actions of an action type terminate the analytic processing.

  For example, if the action is a transfer from an application page to another page, this terminates the framework processing that was triggered by a trigger point associated with the original page.

---

**Note.** A terminal action type is not combinable.

---

- Whether the action can be combined with other actions at runtime. Referred to as a *combinable* action.

  For example, the display alert action type can be combinable; therefore, when two display alert actions are specified, they are combined , and a single popup window appears that includes both alerts in the popup.

- Whether the action is part of an action bundle.

  If actions of *different* action types are combinable with each other, the action types must be included in an action type bundle. Consequently, if combinable action types are not combinable with certain other action types, then the action type does not need to be part of an action type bundle.

- Triggering environment—where the action can be deployed.

  For example, the display alert action executes only from an application page, not from an Application Engine program.

- The trigger types and trigger points that include the action type as a valid action type.

  For example, you may want an action to be valid for a ComponentPostBuild trigger type, specifically the *When a Customer is Presented* trigger point, associated to this trigger type.

# Understanding How Actions Execute

After the framework evaluates conditions for all a trigger point's policies, the actions associated with the policies having true conditions are forwarded to the action framework. The action framework is responsible for firing the actions.

Only one terminal action can be fired for any trigger point. Because a trigger point may be associated with one or more policies, and each policy may include more than one action, it is possible that more than one terminal action is associated with a trigger point. In this case, the first terminal action in the first policy that evaluates to true is executed. If present, a terminal action fires after executing all the non-terminal actions.

Individual action types determine how actions can be combined. Some action types, such as the display alert, combine all their actions from the same trigger point. Therefore at runtime, all the display items appear in the same popup window.

---

**Note.** In order that display actions execute, popup blockers must be turned off.

---

An action that is not combinable will not affect the execution of another action.

 Copyright © 1992-2006, Oracle. All rights reserved.

# Registering Action Types and Action Type Bundles

This section describes the components used to register action types and action bundles.

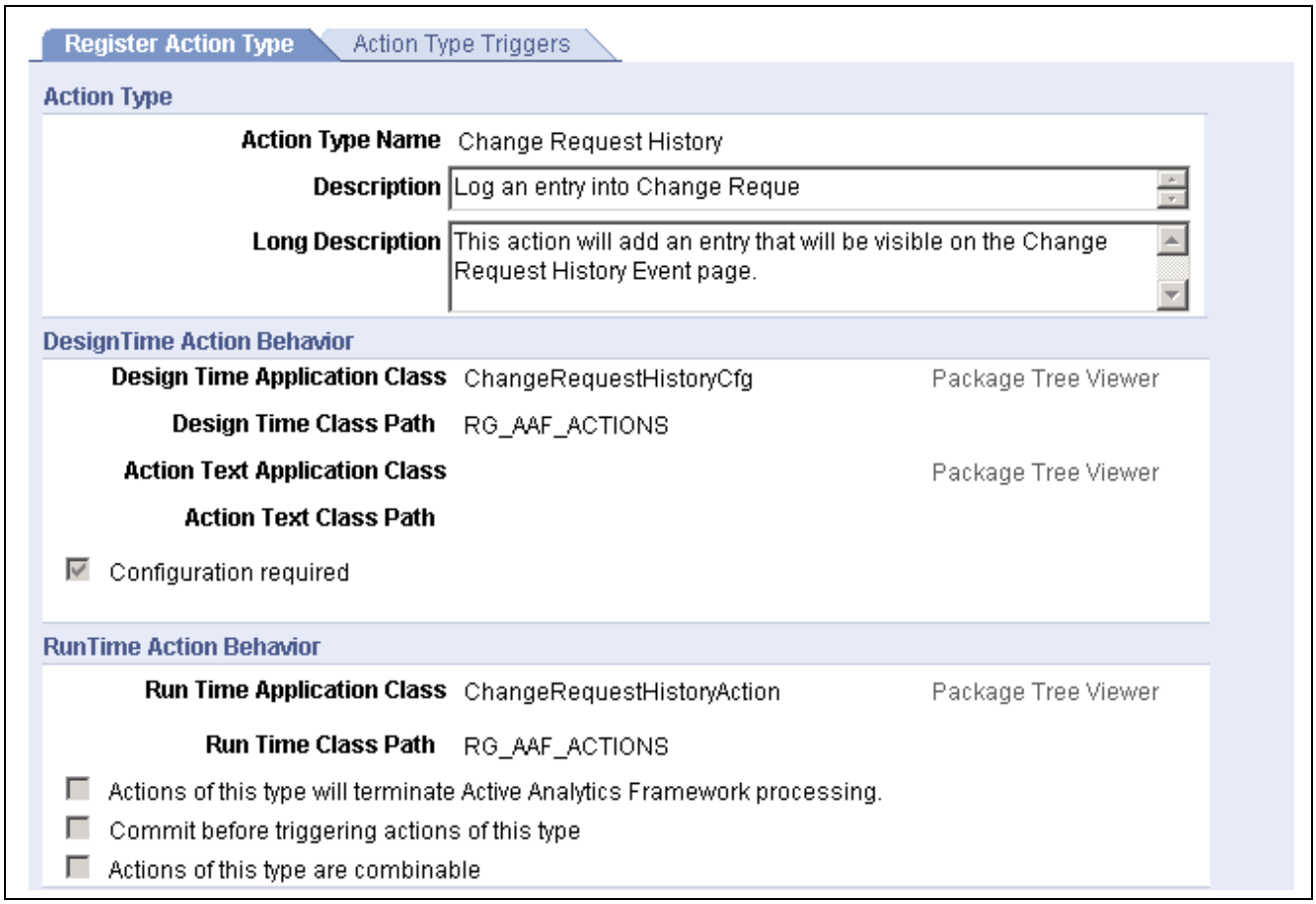
## Pages Used in This Chapter

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Register Action Type | EOCF_ACTN_TYPE_REG | Enterprise Components, Active Analytics Framework, Action Framework, Register Action Type | Registers an action type. |
| Action Type Triggers | EOCF_ACT_TYP_EVNTS | Enterprise Components, Active Analytics Framework, Action Framework, Register Action Type, Action Type Triggers | Specifies action type triggers. |
| Register Action Type Bundle | EOCF_ACTION_BUNDLE | Enterprise Components, Active Analytics Framework, Action Framework, Register Action Type Bundle | Defines bundles of combinable action types. |

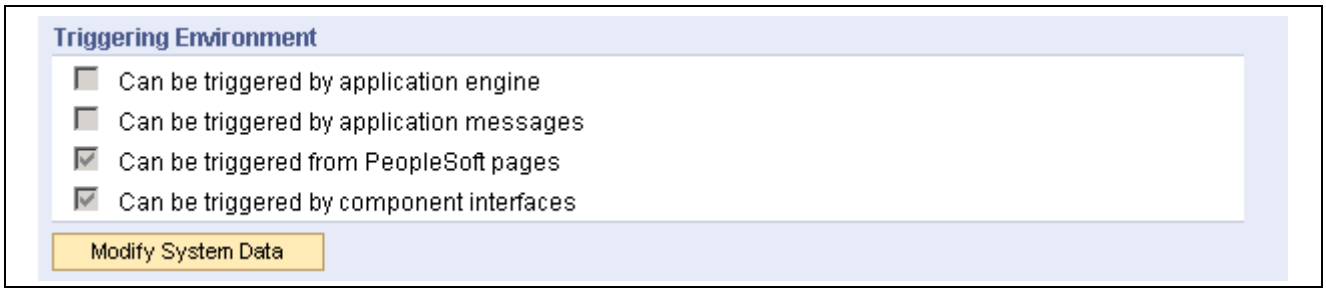
## Registering Action Types

This section describes how to register action types. Access the Register Action Type page.

Copyright © 1992-2006, Oracle. All rights reserved.




Register Action Type page (1 of 2)




Register Action Type page (2 of 2)

**Action Type Name** Name of a class of similar actions.

**DesignTime Action Behavior** Specify the design time details of the action type. When adding actions of this action type in a policy, these design time specifications are used to present the action type configuration page and store the configuration.

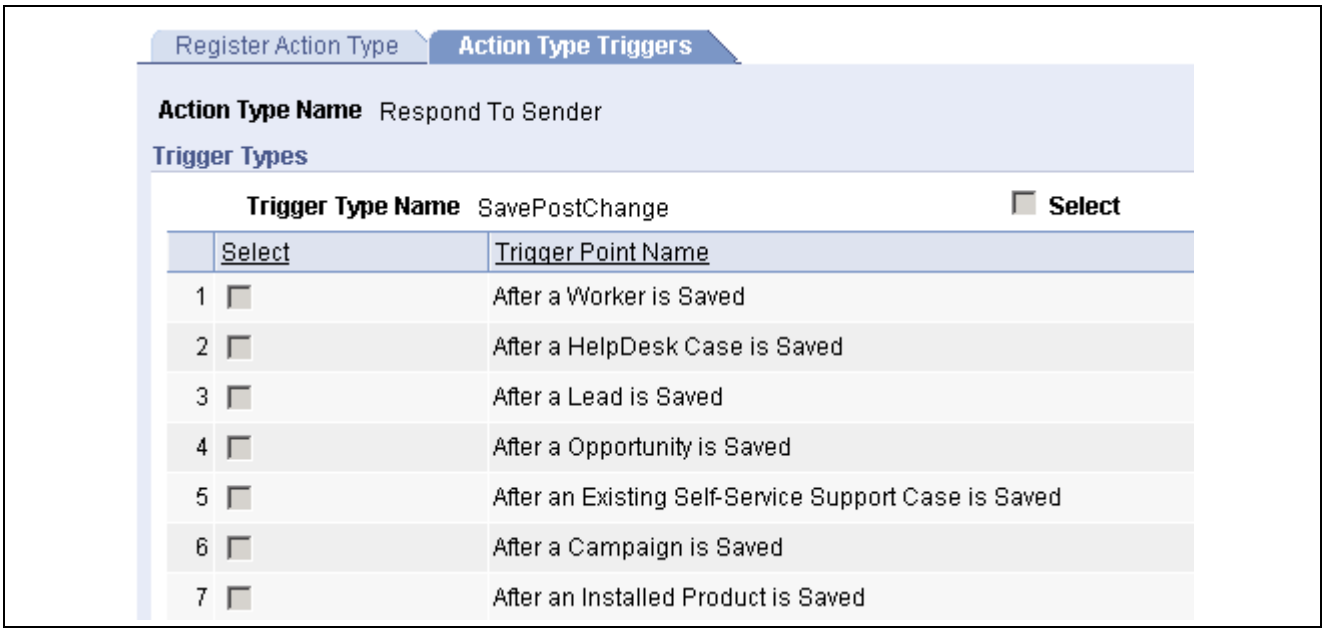
**RunTime Action Behavior** Specify the runtime details of the action type. The application class details specified here are executed at runtime to trigger actions of this type.

**Triggering Environment** Specify the triggering environments to be supported for this action type.

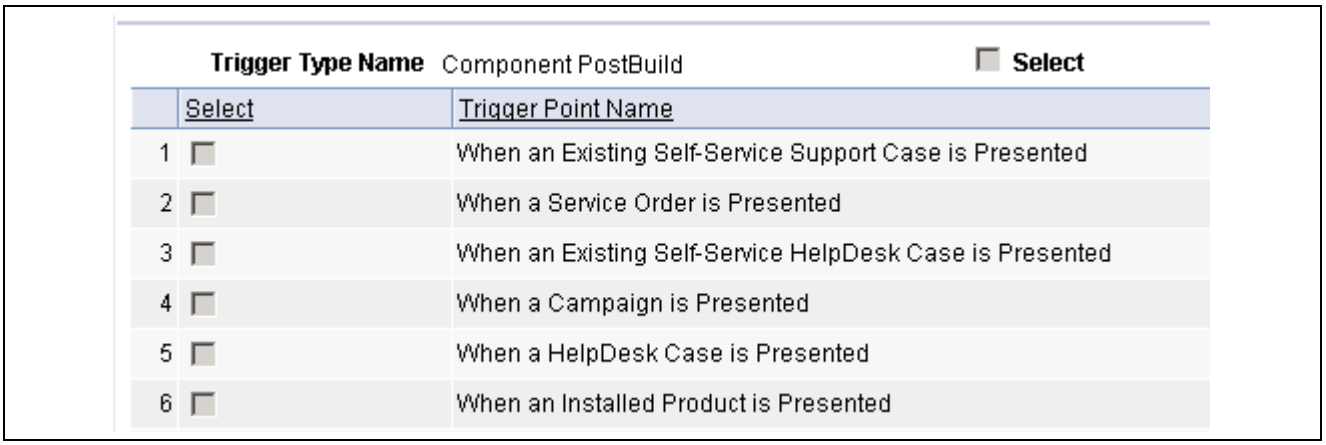
## Registering Action Type Triggers

Access the Action Type Trigger page.


Copyright © 1992-2006, Oracle. All rights reserved.




Action Type Triggers page (1 of 2)




Action Type Triggers (2 of 2)

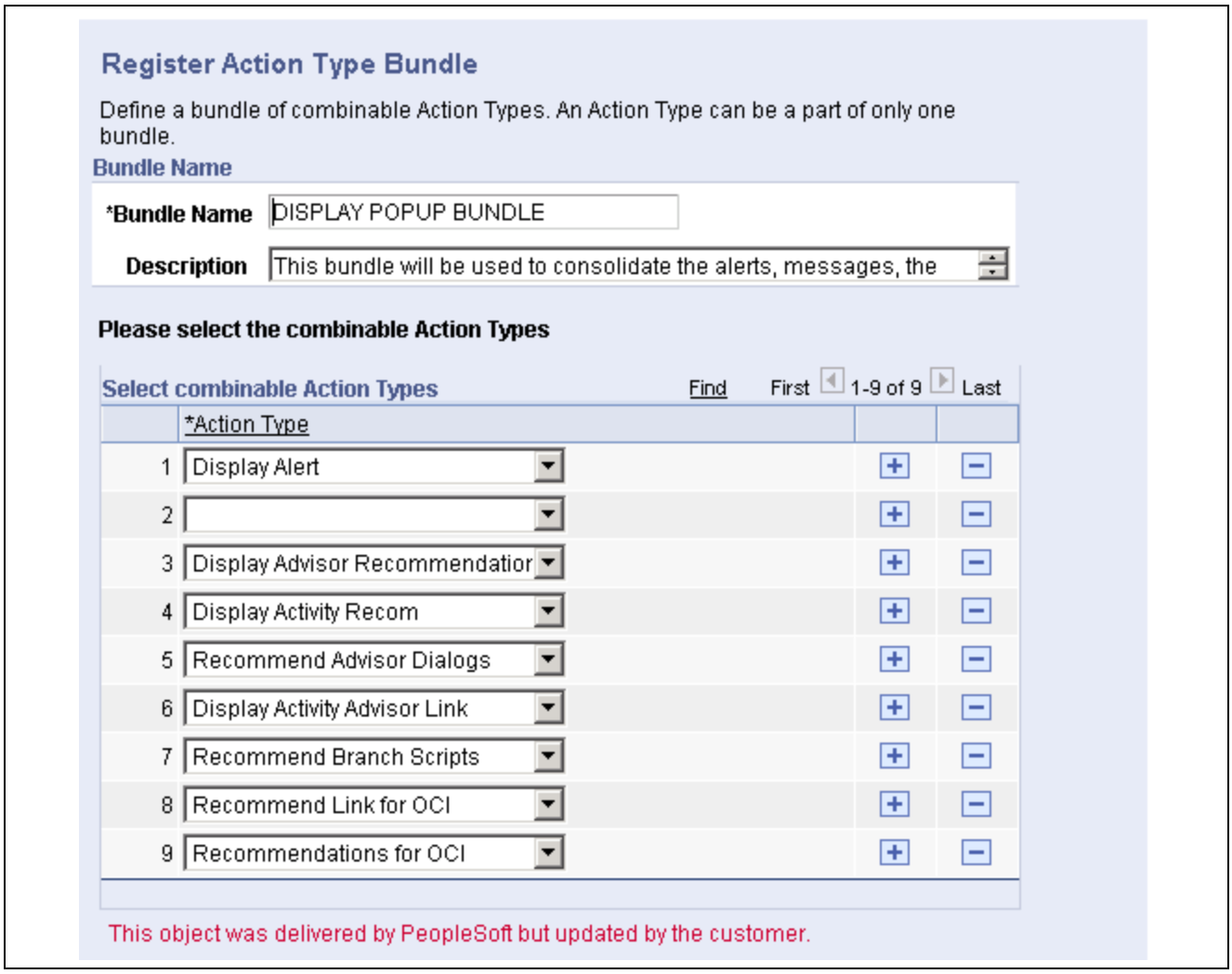
Select the appropriate checkboxes to associate this action type with listed trigger types and trigger points.

- Selecting a trigger type makes this action type available for the selected trigger type.
- One or more trigger points may be selected only if the corresponding trigger type is selected.

## Registering Action Type Bundles

This section describes how to register action type bundles. Access the Register Action Type Bundle page.

Copyright © 1992-2006, Oracle. All rights reserved.




Register Action Type Bundle page

Enter a name and description for this action type bundle. Select the action types that can be combined from the dropdown list in each row.


Copyright © 1992-2006, Oracle. All rights reserved.

# CHAPTER 13

# Administering the Framework

This chapter discusses how to:

- Set logging options for the data library.
- Set installation options.
- Register trigger types and trigger points.
- Define subject areas.
- Register operators and operator sets.
- Register resolution methods.
- Register business domains and categories.
- Register action objectives.

## Using the Setup Components

This section lists the components used for administering the PeopleSoft Active Analytics Framework.

### Pages Used in This Chapter

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Data Library Logging | EOCF_DL_LOG_SET | Enterprise Components, Active Analytics Framework, Setup, Data Library Logging | Used to set Data Library logging options. |
| Install Options | EOCF_INSTALL | Enterprise Components, Active Analytics Framework, Setup, Install Options | Used to set Active Analytics Framework installation options. |
| Register Trigger Type | EOCF_EVTYP_DEFN | Enterprise Components, Active Analytics Framework, Setup, Register Trigger Type | Used to define trigger types. |
| Register Trigger Point | EOCF_EVENT_DEFN | Enterprise Components, Active Analytics Framework, Setup, Register Trigger Point | Used to define trigger points. |

Copyright © 1992-2006, Oracle. All rights reserved.

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Define Subject Area | EOCF_SUBJ_HIER | Enterprise Components, Active Analytics Framework, Setup, Define Subject Area | Used to define subject areas. |
| Register Operators | EOCF_OPERATOR_DEFN | Enterprise Components, Active Analytics Framework, Setup, Register Operators | Used to define operators. |
| Register Operator Sets | EOCF_OPERSET_DEFN | Enterprise Components, Active Analytics Framework, Setup, Register Operator Sets | Used to define operator sets. |
| Register Resolution Method | EOCF_DEF_RESLMTHD | Enterprise Components, Active Analytics Framework, Setup, Register Resolution Method | Used to define the resolution method. |
| Register Business Domain | EOCF_BUS_DOMAIN | Enterprise Components, Active Analytics Framework, Setup, Register Business Domain | Used to define a business domain, which is used to functionally classify trigger points. |
| Register Category | EOCF_IACATEGORY | Enterprise Components, Active Analytics Framework, Setup, Register Category | Used to functionally classify policies. |
| Register Action Objective | EOCF_ACTION_OBJ | Enterprise Components, Active Analytics Framework, Action Framework, Register Action Objective | Used to functionally categorize actions defined in policies. |

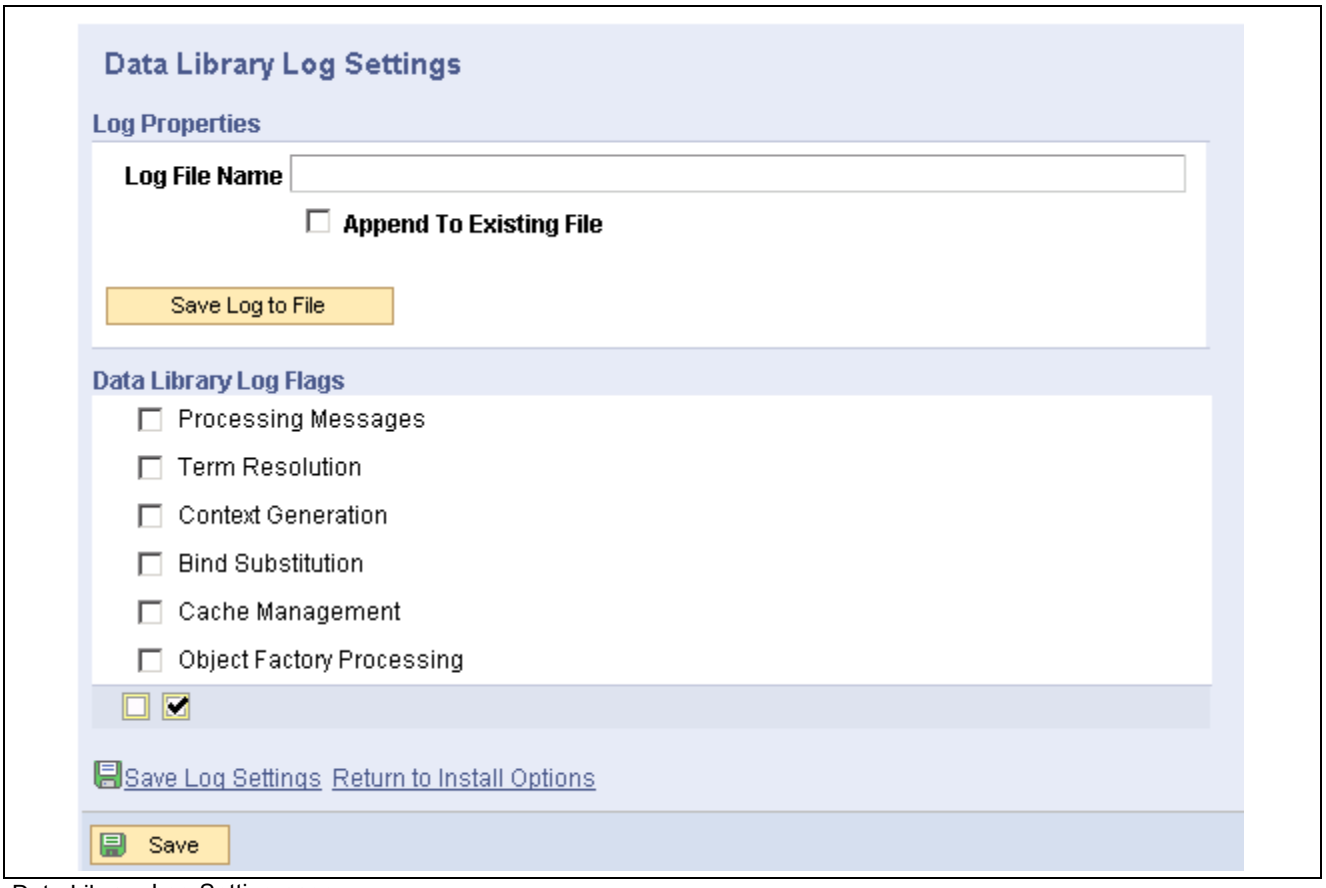
# Setting Log and Installation Options

This section describes the logging and installation options

## Setting Data Library Log Options

Access the Data Library Log Settings page.

 Copyright © 1992-2006, Oracle. All rights reserved.



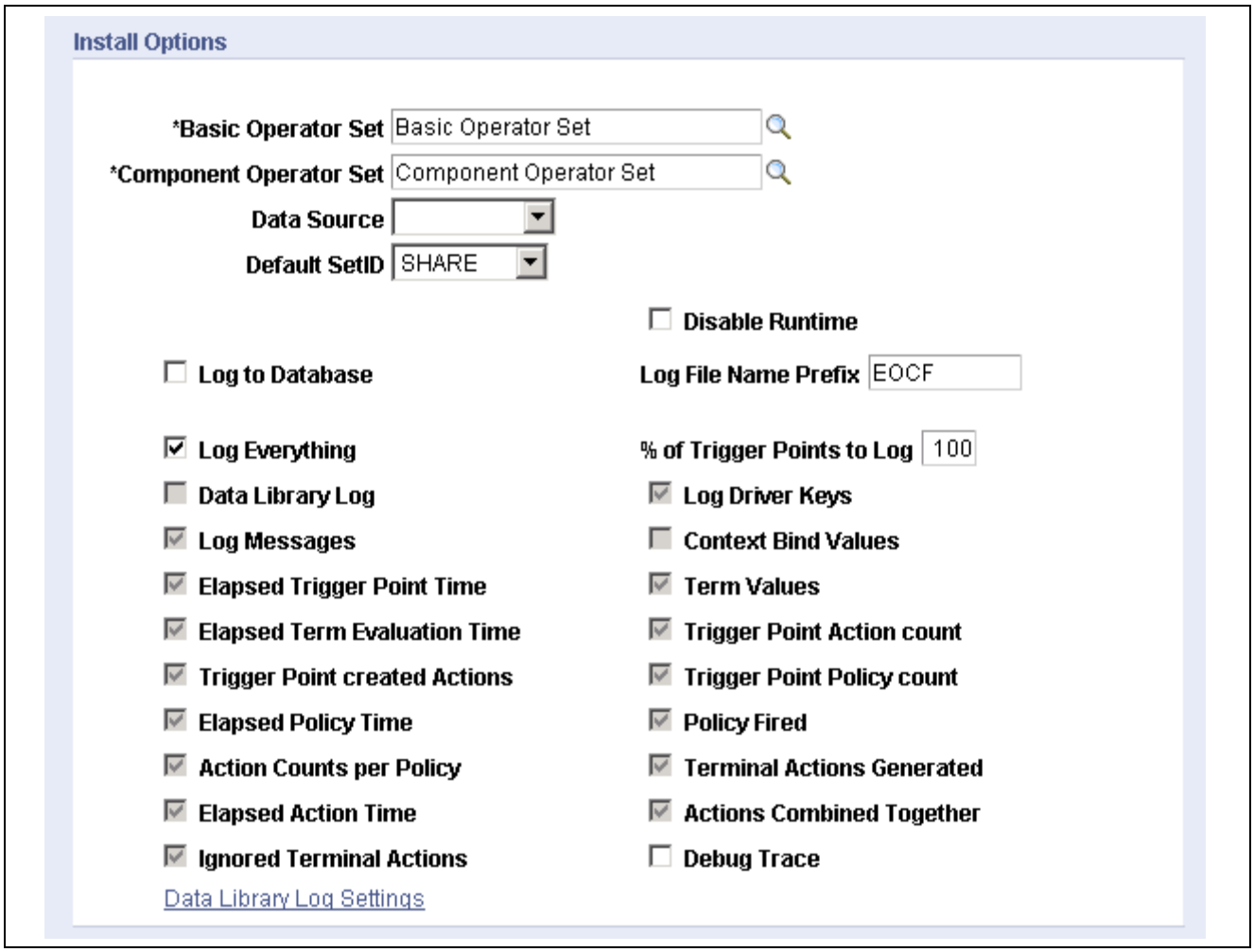

Data Library Log Setting page

| | |
|---|---|
| **Log File Name** | Enter a name for the log file and select Append to an Existing File, if desired. |
| **Term Resolution** | Log technical details of resolution of individual terms. |
| **Context Generation** | Log technical details of the generation of contexts. |
| **Bind Substitution** | Log technical details of implementation binds substituted with values from the runtime context. |
| **Cache Management** | Log technical details of usage of cache. |
| **Object Factory Processing** | Log technical details of the loading of framework object definitions. |

## Installation Options

Access the Install Option page.

Copyright © 1992-2006, Oracle. All rights reserved.




Install Options page

**Data Source** The product line for this database, for example CRM or HMS.

**Default SetID** The setID value to use for a component that does not use set control

**% of Trigger Points to Log** When logging for performance monitoring, the impact of logging can be alleviated by only logging a percent of trigger points executed. Specify a percent value between 1 and 100.

**Disable Runtime** This option disables all trigger points from running.

**Log to Database** Determines if the log will be written to the database, or to a text file.

**Log File Name Prefix** If a file is logged instead of the database, this prefix is prepended to each user's log file.

---

**Note.** The decision engine log file is created in the default files directory for this application server instance. Typically, the application server uses the directory %PS_HOME%\appserv\<servername>\files. The application server creates this directory as needed; therefore, if no log files have been written, it may not yet exist.

---

**Log Everything** If checked, causes all normal logging options are activated.

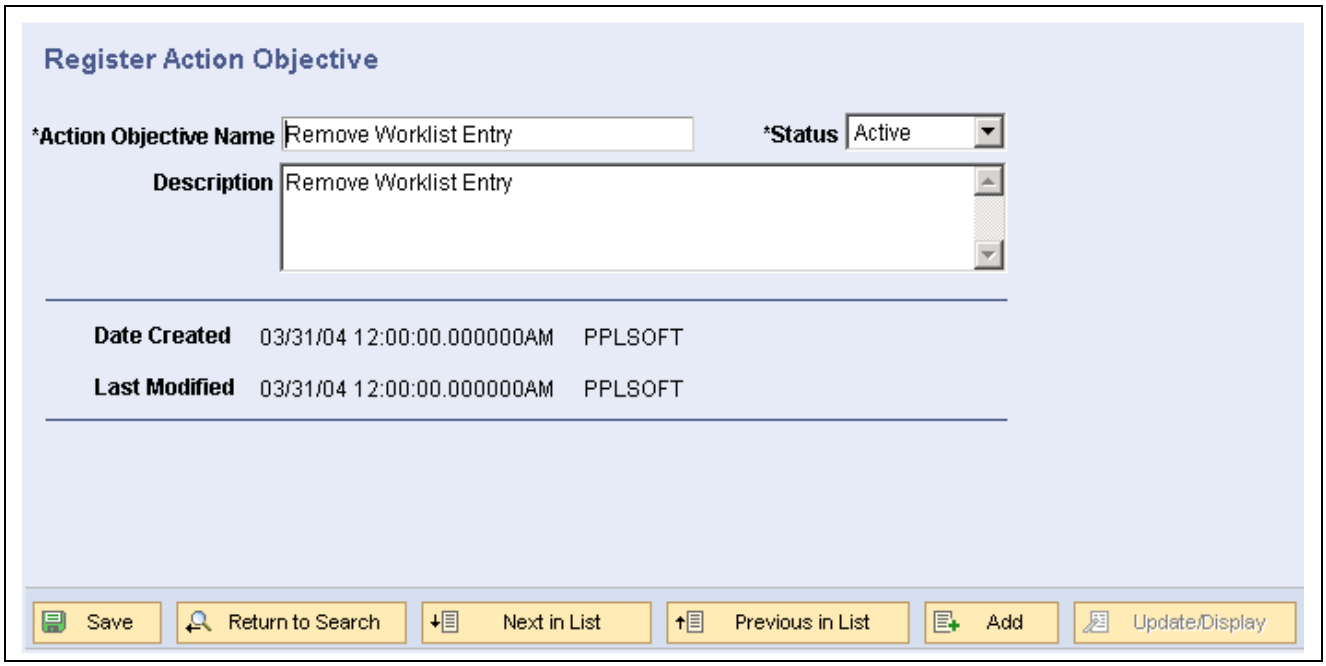
 Copyright © 1992-2006, Oracle. All rights reserved.

| | |
|---|---|
| **Data Library Logging** | Includes the data library log entries in the decision engine log. |
| **Log Driver Keys** | Log the primary key values driving each trigger point's context (transaction). |
| **Log Messages** | Log messages that policies or actions may generate. |
| **Context Bind Values** | Log values retrieved directly from the operant context. |
| **Elapsed Trigger Point Time** | Log the time required to complete a trigger point. |
| **Elapsed Term Evaluation Time** | Log the time required to evaluate a term. |
| **Term Values** | Log the values of evaluated data library terms. |
| **Trigger Point Action Count** | Log the number of actions created by the execution of a trigger point. |
| **Trigger Point Policy Count** | Log the number of policies evaluated for a trigger point. |
| **Trigger Point created Actions** | Log the actions created during the evaluation of policy conditions for a trigger point. |
| **Elapsed Policy Time** | Log the time required to evaluate the condition portion of a policy. |
| **Policy Fired** | Log whether a policy condition was true or false. |
| **Action Counts per Policy** | Log the number of actions created by a policy. |
| **Terminal Actions Generated** | Log terminal actions created by a policy. |
| **Elapsed Action Time** | Log the time it takes an action to complete. |
| **Actions Combined Together** | Log actions that were combined during execution. |
| **Ignored Terminal Actions** | Log any terminal actions that could not be invoked. |
| **Debug Trace** | Log debugging information, not normally used. |

# Registering Action Objective

Access the Register Action Objective page.

Copyright © 1992-2006, Oracle. All rights reserved.




Register Action Objective page

Enter the Action Objective Name and a description.

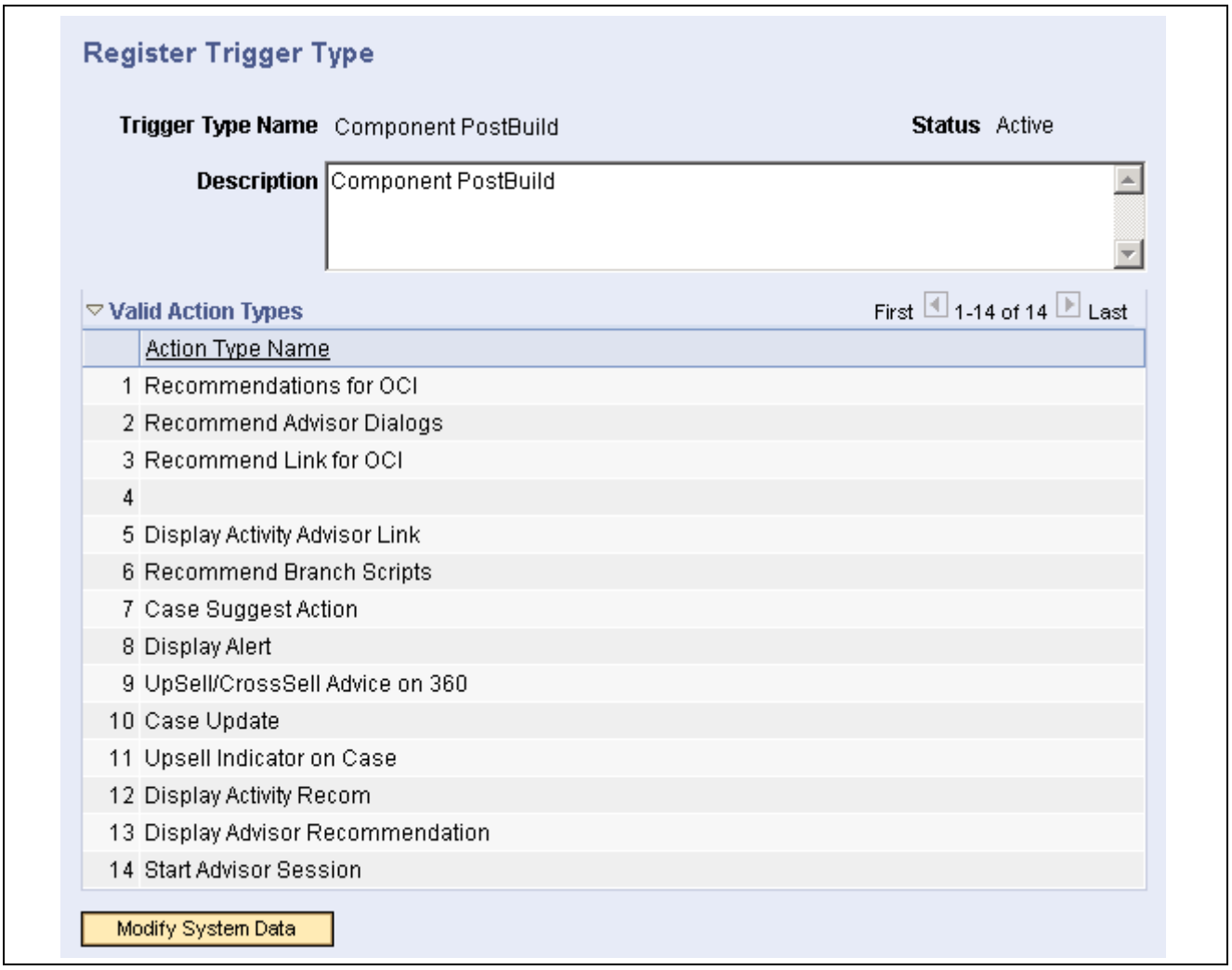
# Registering Trigger Types and Trigger Points

This section describes the components used to register trigger types and trigger points.

## Registering Trigger Types

Access the Register Trigger Type page.


Copyright © 1992-2006, Oracle. All rights reserved.



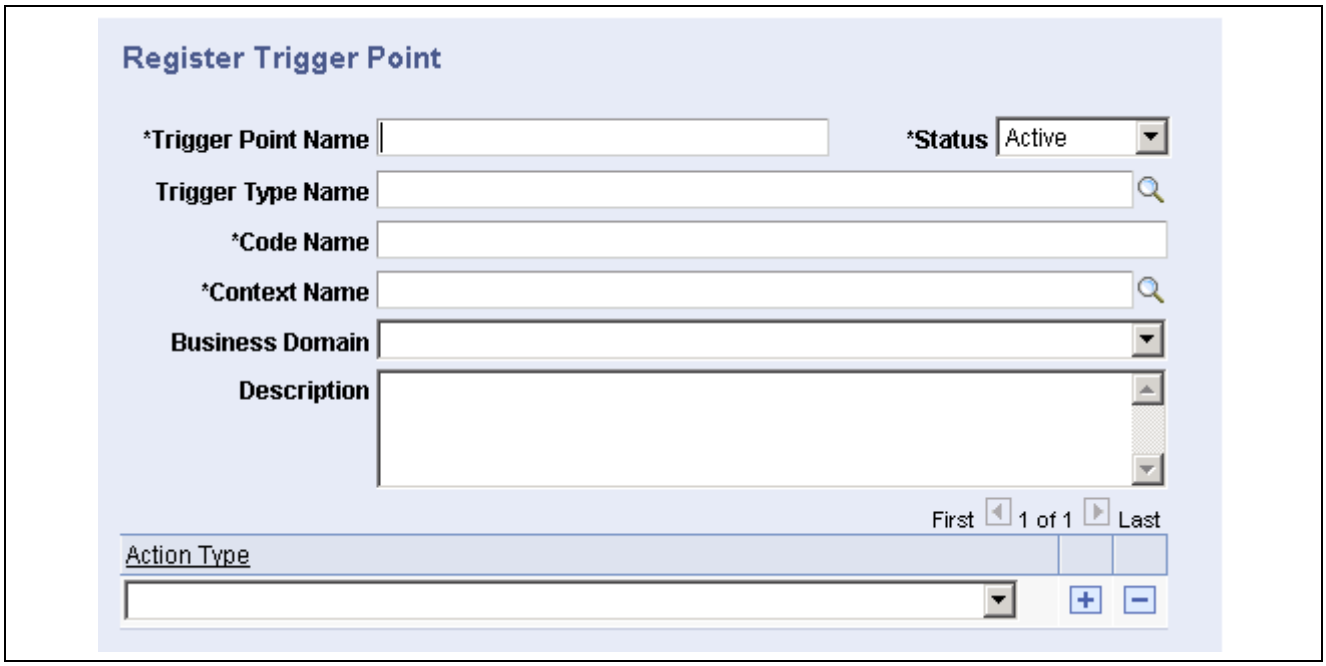

Register Trigger Type page

Enter details about the trigger type you are defining and select the valid actions.

## Registering Trigger Points

Access the Register Trigger Point page.

Copyright © 1992-2006, Oracle. All rights reserved.




Register Trigger Point page

**Trigger Point Name** Unique and descriptive name of the trigger point.

**Trigger Type** Used by IT developers to figure out which PeopleSoft event in the component needs to be enabled with this trigger point. Also restricts the list of valid action types that may be used when creating policies for this trigger point.

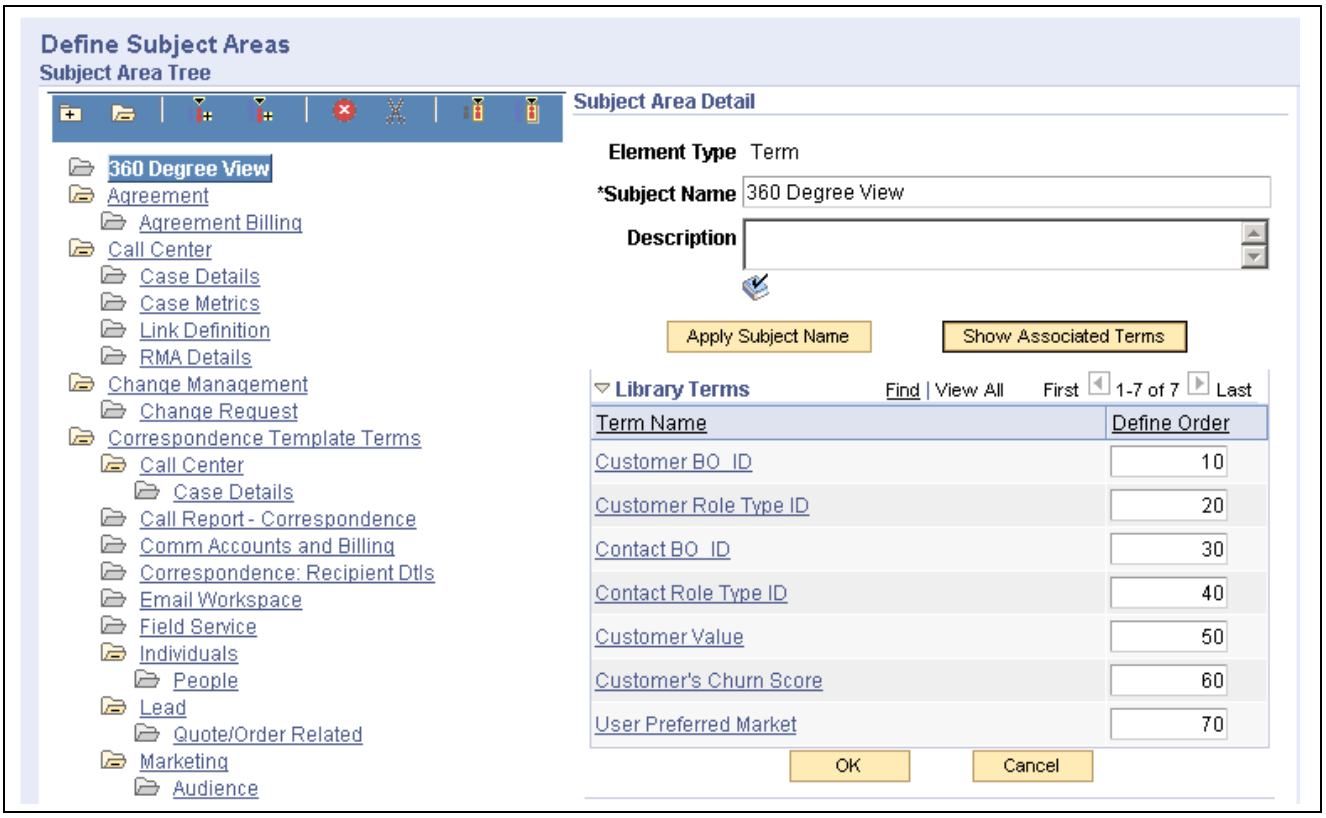
**Code Name** Used to reference the trigger point when programmatically enabling this trigger point for a PeopleSoft component. Changes made to the code name, after enabling this trigger point for a PeopleSoft component, may disrupt the execution of policies for this trigger point at runtime.

**Context Name** Defines the context for the trigger point.

# Defining Subject Areas

Access the Define Subject Area page.


Copyright © 1992-2006, Oracle. All rights reserved.




Define Subject Areas page

**Subject Area Detail**

Enter a Subject Name and Description and click Apply Subject Name. The subject area name may not be unique.

Click Show Associated Terms to display the list of terms associated to this subject area node. Order the terms appearing for a subject area node by specifying display order numbers. The order depicts how the terms are displayed for a subject area node in the term list presented in the condition builder.

**Subject Area Tree**

The subject area tree displayed on the left-hand side has the following icons:

- *Expand All*. Expands all subject area nodes.
- *Collapse All*. Collapses all subject area nodes.
- *Add Sibling*. Adds a new sibling node under the subject area node currently selected. Enter a subject area name and description on the right hand pane and click Apply Subject Name to apply changes.
- *Add Child*. Adds a new child node for the subject area node currently selected. Enter a subject area name and description on the right-hand side and click Apply Subject Name to apply changes.
- *Delete Node*. Deletes the selected node.
- *Cut*. Cuts the currently selected node.
- *Paste as Sibling*. Pastes the node previously cut as a sibling of the currently selected node.

Copyright © 1992-2006, Oracle. All rights reserved.

- *Paste as Child*. Pastes the node previously cut as a child of the currently selected node.

# Registering Operators and Operator Sets

This section describes the components used to register operators and operator sets.

Operators within PeopleSoft Active Analytics Framework are defined in text expressions that are used to evaluate a condition at runtime. The operand values in the text expressions must be substituted with an operator definition for integration with policies. The substitution of operand values is ensured by the correct placement of meta-text corresponding to the desired operand or source term.

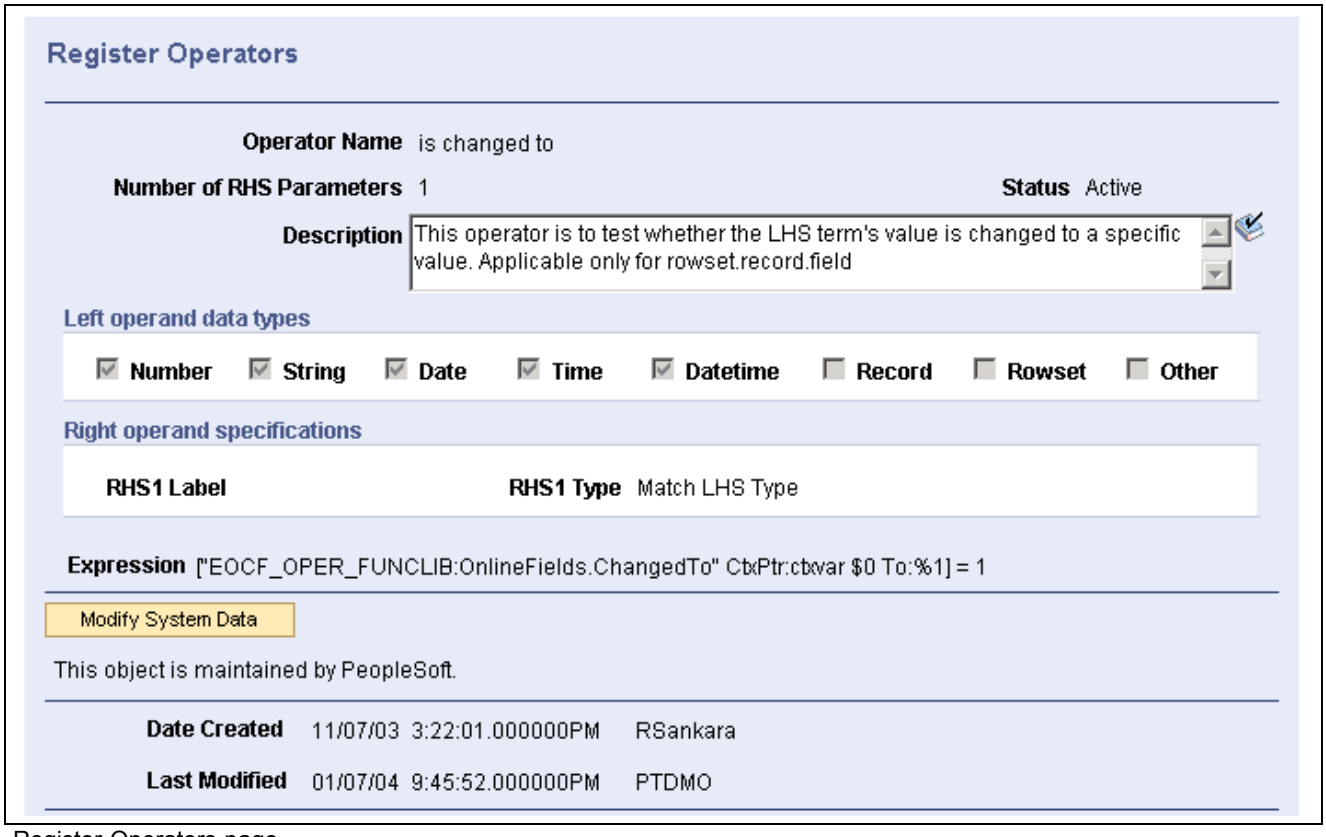
To refer to the left-hand-side of a condition, use %0 as the value. To refer to right-hand side operands 1–4 , use the text values %1, %2, %3 , and %4, respectively. For example, the following operator expression determines if the numeric term's value is less than a right-hand side value:

%0 < %1

**Note.** If you edit operators, policies that reference them must be recompiled. Do this by editing and saving the policy.

## Registering Operators

Access the Register Operators page.




Register Operators page


Copyright © 1992-2006, Oracle. All rights reserved.

Register custom operators by defining the left-hand side and right-hand side operand specifications and entering expression text.

**Operator Name** Enter the name of the operator you are defining. Operator names may not be unique as long as the expression texts are different.

**Number of RHS Parameters** Specify the number of right-hand side parameters that are required to evaluate the boolean value for this operator. The condition builder uses this to display one or more right-hand side fields to enter values.

**Note.** In this release, the condition builder interface supports a maximum of 2 right-hand side parameters for an operator.

**Left operand data types** Specify the supported data types for the left-hand side operand of this operator. This data is used to constrain the list of operators in the condition builder.

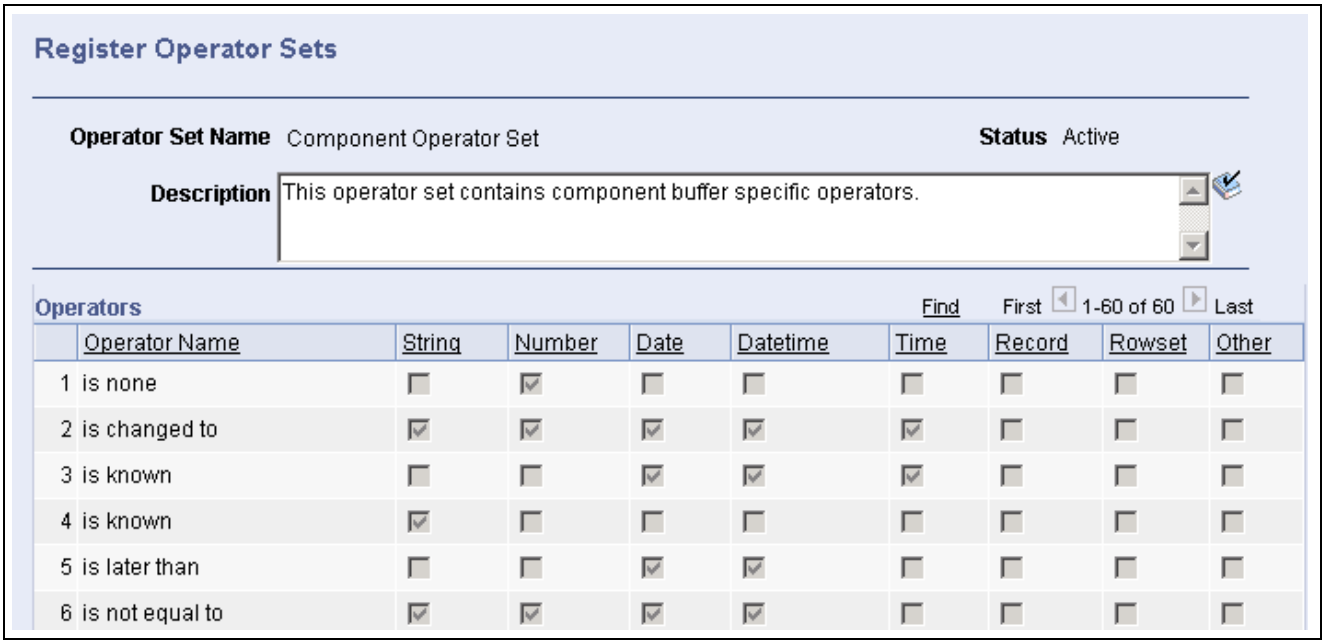
**Right operand specifications** Specify the type for each of the right-hand side parameters.

- *Match LHS Type*. Supports the left-hand operand data types.
- *Fixed Type*. Specify a fixed data type for this parameter.
- *Multi-Select*. Enables selection of multiple values for this parameter at condition building time.

**Expression** Specify the technical expression defining how the operator works.

# Registering Operator Sets

Access the Register Operator Set page.




Register Operator Sets page

Operator sets define the list of valid operators for a term selected on the left-hand side of a condition.

Enter a unique Operator Set Name and description, and add valid operators for this set.

Copyright © 1992-2006, Oracle. All rights reserved.

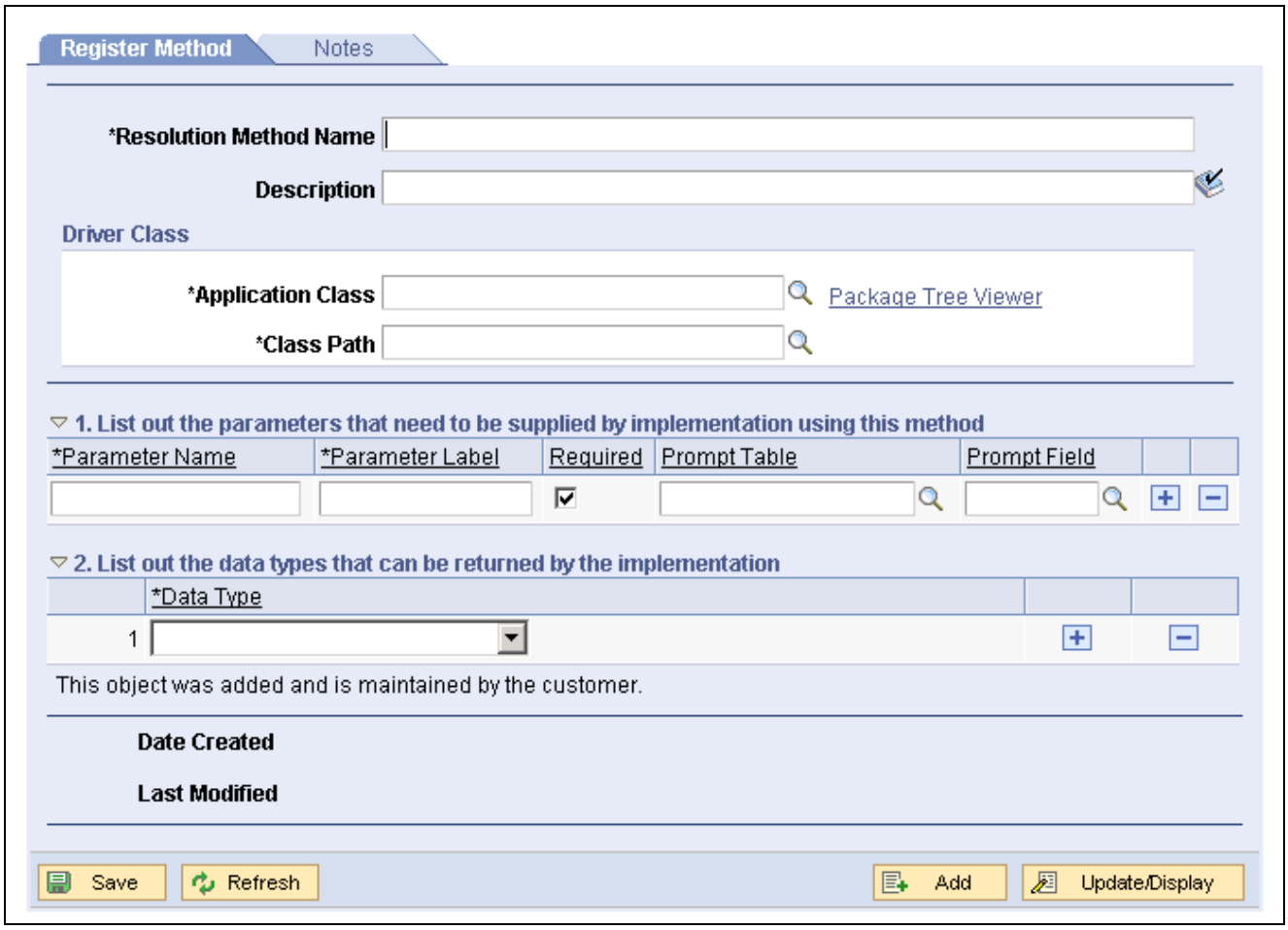
# Registering Resolution Methods

This section describes how to register a resolution method.

Resolution methods are used to build implementations. PeopleSoft supports the following resolution methods:

- Application class.
- PS Query.
- SQL Object.
- Record.Field.

Access the Register Resolution Method page.




Register Method page

**Resolution Method Name** Unique name of the resolution method.

**Driver Class** The application class that encapsulates the data retrieval mechanism for this resolution method.

**Parameters...** Lists the parameters that an implementation using this method needs to supply.

**Datatypes...** Lists the valid data types that can be returned by an implementation using this resolution method.


Copyright © 1992-2006, Oracle. All rights reserved.

# Registering Business Domain and Category

This section describes how to register a business domain and how to register a category.

## Register Business Domain

Access the Register Business Domain page.

Enter the Business Domain name and Description and select the appropriate status.

## Register Category

Access the Register Category page

Enter the Category Name and Description and select the appropriate status.

Copyright © 1992-2006, Oracle. All rights reserved.

 Copyright © 1992-2006, Oracle. All rights reserved.

# CHAPTER 14

# Considerations for Enabling the Framework

This chapter discusses considerations when:

- Creating contexts.
- Creating custom operator expressions.
- Creating a trigger point.
- Enabling a trigger point in a PeopleCode event.
- Enabling a display action.
- Creating a new action type.
- Creating an application class implementation.
- Creating a PS query implementation.

## Considerations When Creating Contexts

To create an online context, you must be able to specify a component interface. You may encounter situations where sets of components share the same base records and display the same information onscreen, however for different purposes. In this case, decide if you need to create two contexts or if you can use one.

### Whether to Create One or Two Contexts

Determine if the two contexts' common terms should be registered as the same term objects, or as different term objects. When common terms have the same meaning, it is most economical to re-use the common term definitions between the two contexts.

Create two contexts if:

- The components have actionable fields that they do not share in common.
- One or the other of the components need significant customization.

### Considerations when Exposing Component Buffer Fields as Terms

Carefully consider which component buffer fields to be made available as terms.

- Keep the big picture in mind—by selecting terms that may be reused at some point.

  However, you would not want to expose every component field as a term. Some of the transactional components that allow users to perform complex tasks may have component fields that are used as work fields to implement the component, but which do not represent functional data. Exposing these fields as terms could overpopulate the data library.

Copyright © 1992-2006, Oracle. All rights reserved.

- PeopleSoft recommends that you expose terms of those data elements on the component that are displayed to the end user and the elements that comprise the component's search keys.
- Work-fields may contain valuable data and may be exposed as terms, but if these fields have not yet been populated by the time that the values from the corresponding terms are requested, unpredictable results may occur. If you know for certain that the work-fields will contain data at the appropriate times, such as in the example of a work-field whose data is computed as part of component pre-build, then there is no harm in exposing the data element as a term. Although there is a Manage Context component for adding terms to a context, configuring terms is easier in the Generate Context component.

### Considerations When Naming Terms

Note the following when you name terms:

- The framework prefixes each alias with the name of the context to reduce the creation of duplicate names.
- In record and rowset objects, the term name defaults to the name of the page buffer scroll, which will not be meaningful for the end user. PeopleSoft recommends this name be changed to something that makes sense for the user.
- Generally, rowset objects' names should be plural and record object term names should be singular. For example, if a component's scroll-level 1 contained purchase order line items, the rowset object is named Detail Items and the record object is named Line Item.
- If any terms are filed under alternate subject areas, use the Overridden Subject Area prompt to do so. PeopleSoft recommends that you carefully consider that all the names generated make sense before importing the context. Whenever context terms have the same functional meaning, have the context refer to pre-existing terms rather than creating new ones.

## Considerations When Creating Custom Operator Expressions

Each operator definition contains an *operator expression*, a text template that defines the meaning of the operator and how operands are integrated to the expression. All operator expressions must be boolean expressions.

Comparisons supported in operator expressions are: =, <. >, <=. >=, <> (not equal). All comparisons are type-driven; for example, a number cannot be compared to a string.

If type conversion is necessary, the value to be converted may be preceded by the tokens string, number, date, datetime, or time, as needed.

Values are placed in expression text with two types of substitution tokens, location number and ? (question mark).

### Location Number As An Operand

Location number determines the operand in the expression that is being referred to. For the left-hand side term, the location number is zero; for the right-hand side elements, the range is from one to four, depending on which right-hand side parameter is being referred to. Usually, the location number is preceded by a percent-sign. The percent-sign and location number combination form a complete operand token.

A complete operand token refers to the value of the configured operand. When creating a condition, an expression-text entry will be substituted for the operand-token. This expression-text entry corresponds to the definition of the operand value's source.

 Copyright © 1992-2006, Oracle. All rights reserved.

Sometimes a term ID value needs to be provided in the expression instead of a location number. In this rare case, instead of a percent sign, a dollar-sign is used to precede the location-number.

Parenthesis around sub-expressions are allowed, so long as each left parenthesis has a corresponding right parenthesis, and so long as the sub-expressions denoted within parentheses are valid. (While "(%0) > %1" and "(%0 > %1) are valid, "(%0 >) %1" is not.) Constants are available for the current date, date-time and the current time by means of the items "%today", "%timestamp" and "%now".

### Use of Term Code

In addition to referring to the results of terms that have been configured in the Operator Definition page, references to terms may be embedded directly in the expression text by means of the term code. A reference to a term is made by placing the term code name (which may not contain spaces) within square brackets. The bind parameters needed for the term must be part of the context, therefore the operator should be part of a restricted operator set that's available only where the operator will be known to be valid.

Parameters may also be manually specified for a term by following the term code name with a parameter specification, which consists of a name, a colon symbol, and the value to be bound to the parameter.

In the following example, a term is used as an input to another term in a condition to detect if a customer's car is out of warranty:

[WarrantyExpirationDate VIN:[PrimaryVehicle DriversLicense:%0]] < %today

Generally this type of construction is awkward and difficult to maintain, but it is available if needed. A more maintenance-friendly facility that doesn't require the configuration overhead of a term definition is a member-function call on an application class.

There are two rules that must be satisfied to use this facility:

- The class must not require any constructor arguments.
- The method to be invoked must accept an *array of any* as its only parameter.

To use this facility, set up an operator expression as if you are referring to a term; instead of using a term code name, use a quoted string value giving the fully qualified name of the class to be instantiated followed by a period and the name of the member function to be invoked. For example:

["MyAppPackage:MySubPackage:MyClass.MyMethod" MyParm1:%0 MyParm2:%2] = %1

Should an application class method need to return a boolean result, the standard practice is to return a number, and to compare the number to 0 for false values or to 1 for true values.

---

**Note.** Be aware that when you create operators for a specific purpose, they are neither valid nor relevant except in a specific situation. Therefore, the operator should belong to an operator set that corresponds to the context or term for which the custom operator makes sense.

---

# Considerations When Creating a Trigger Point

Use caution when introducing new trigger points, especially in the FieldChange PeopleTools event. These trigger points, depending on where they are set, could result in significant overhead to the system.

In addition, introducing a new trigger point increases the likelihood of adding policies. These policies, depending on the number of terms and the mode of retrieval, could increase system overhead. This can result in a unfavorable user experience and throughput.

Copyright © 1992-2006, Oracle. All rights reserved.

PeopleSoft recommends that FieldChange event trigger points be introduced only when it is critical. Consider deferring the execution of policies to a Save event.

# Enabling a Trigger Point in a PeopleCode Event

The events in which a trigger point is to be executed must include PeopleCode. The following is a minimal example:

```
Declare Function GetRuleService PeopleCode FUNCLIB_EOCF.EOCF_DE_PUBLIC Field⇒
Formula;  GetRuleService().reset.SET_CONTROL = &mySetControlValue;
GetRule Service.fireEvent(My_Trigger_Point_Code_Name);
```

**Note.** The use of the reset property on the first line is critical to the framework's correct functioning. If reset is not used, the state of the RuleService (the runtime API for triggering the decision engine) is unknown from previously executed PeopleCode.

# Enabling the Display Action in a PeopleSoft Application Page

To enable the provided display alert action type:

- Add the PeopleTools sub-page EOCF_DISPLAY_SUBPG at Level 0 to each page within the component.
- Ensure that no other fields overlap the read-only display alert sub-page, otherwise the display alert action will not work correctly.
- Ensure that the roles having access to the transactional component that requests the framework to evaluate policies, have access to EOCF9002 permission list.
- Ensure that all popup blockers have been turned off.

**Note.** Any display action types created must be registered in the display action type bundle in addition to the delivered display alert action type.

# Considerations When Creating a New Action Type

When creating a new action type, consider the following:

- Location of the code that handles the design-time and runtime aspects of the action type.
- Configuration requirements of the new action type before being enabled.
- Whether or not the action type is a terminal action.

  For example, if the action is to transfer to another page, this terminates framework processing. If there are several terminal actions associated with a trigger point, remember only the first of the terminal actions will be chosen, and it fires after all other non-terminal actions have been fired.

## Configuring the Action Type

If your action type requires a configuration page, perform the following:

- Create record definition(s), keyed by EOCF_ACTION_ID, to capture design time configuration data.

 Copyright © 1992-2006, Oracle. All rights reserved.

- Create an action configuration component and page.
- Insert EOCF_ACTN_CFG_SBPG in the configuration page. This displays the action type and action name on the configuration page.
- Insert the policy information subpage, EOCF_RULEINFO_SBP, on the page. This displays policy information including the policy name and condition text.
- If you need to access terms, put a clone of EOCF_SRCH_DSP_AL as a secondary page on your action configuration page; and add the following code in the page's activate. (See the display alert action configuration page, EOCF_DSPL_ALRT_CFG, as an example of a configuration page)

```
import EOCF_CLF_RB:Definition:Rule:Rule;
import EOCF_CLF_RB:UI:*;

Component SearchBuilder &cobjSearchBuilder;
Component SearchConfig &cobjSearchConfig;
Global Rule &gobjRule;
Local RuleBuilder &objRuleBuilder = create RuleBuilder();
 &cobjSearchConfig = create EOCF_CLF_RB:UI:SearchConfig();

/*** Build Search page display **/
/*** Add where clause to filter terms by scalar **/
&cobjSearchConfig.AddCustomWhereClause(FetchSQL(SQL.EOCF_WHERECL_TMSCALAR), 0,
   "");
&cobjSearchBuilder = create SearchBuilder(EOCF_ACTION_WRK.EOCF_CONTEXT_ID,
   &cobj SearchConfig);
&cobjSearchBuilder.BuildDisplay();
/*** Populate Rule info **/
&objRuleBuilder.PopulateRuleInfo(&gobjRule);
```

- Insert a push button on the configuration page to transfer to this secondary page (for the purpose of choosing a term).

  Some of the terms on your configuration page may be configurable. Therefore, you must paste another subpage, EOCF_TERMCFG_SBP, on the configuration page for this purpose.
- Insert the following page activate code on your action configuration page to initialize the term configuration subpage:

```
import EOCF_CLF_RB:Definition:Rule:Rule
import EOCF_CLF_RB:UI:*;
import EOCF_CLF_RB:UI:ConfigBuilder;
import EOCF_CLF_RB:Definition:RuleTermConfig;
import EOCF_CLF_DL:Factory:DataLibraryFactory;
import EOCF_CLF_DL:Definition:Term:TermDefn;
import EOCF_CLF_DL:Utility:DLConstants;
Local Rowset &rs_level0, &rsActionTypeDefn;
Local number &bundleId, &actionId;
Local number &actionTypeId;
Local string &isTerminal, &isCombinable, &isConfigurable;
Local string &aeTriggered, &appMsgTriggered, &ciTriggered, &piaTriggered;
Local string &commit, &path, &id, &dtAppClassId, &appClsPath, &dtAppClsPath,
   &rtApp ClassId, &rtAppClsPath, &actTypeName, &actionName;
```

Copyright © 1992-2006, Oracle. All rights reserved. 

```
Global EOCF_CLF_RB:Definition:Rule:Rule &gobjRule;
Local EOCF_CLF_RB:UI:RuleBuilder &objRuleBuilder = create EOCF_CLF_RB:UI:
   Rule Builder();
Local EOCF_CLF_RB:Definition:RuleTermConfig &objRuleTermConfig = create
   EOCF_CLF_RB: Definition:RuleTermConfig();
Local EOCF_CLF_DL:Factory:DataLibraryFactory &objDlFactory = create
   EOCF_CLF_DL: Factory:DataLibraryFactory();
Local EOCF_CLF_DL:Definition:Term:TermDefn &objTermDefn = create
   EOCF_CLF_DL: Definition:Term:TermDefn();
Local EOCF_CLF_DL:Utility:DLConstants &objDLConstants = create EOCF_CLF_DL:
   Utility: DLConstants();
/*********************************************************************/
/* CREATE A CONFIG BUILDER TO CREATE UI ELEMENTS THAT ENABLE TERM    */
/* CONFIGURATION-TO BE DONE ONLY BY ACTIONS THAT NEED TO ACCESS DATA */
/* LIBRARY TERMS, AND THUS HAVE THE TERM-PICKER SCROLL AND THE TERM  */
/* CONFIGURATION SUBPAGE ON THE PAGE */

Local ConfigBuilder &objConfigBuilder;
Local Rowset &rs1 = GetLevel0()(1).GetRowset(Scroll.EOCF_DS_ALT_TRM);
&objConfigBuilder = create ConfigBuilder();
&objConfigBuilder.InitDisplay();
For &i = 1 To &rs1.ActiveRowCount
   &termId = &rs1(&i).EOCF_DS_ALT_TRM.EOCF_LIB_TERM_ID.Value;
   If All(&termId) Then
      If All(&rs1(&i).EOCF_DS_ALT_TRM.EOCF_CONFIG_ID.Value) Then
         /*******************************/
         &objRuleTermConfig = create EOCF_CLF_RB:Definition:RuleTermConfig();
         /*******************************/
         &objRuleTermConfig.EOCF_CONFIG_ID = &rs1(&i).EOCF_DS_ALT_TRM.
            EOCF_CONFIG_ ID.Value;
         &objRuleTermConfig.getConfiguredTerm( True);
         &rs1(&i).EOCF_DS_AL2_WRK.EOCF_GOTO_BTN.Label = &objRuleTermConfig.
            Get ConfigTermLabel(EOCF_ACTION_WRK.SETID.Value);
         &rs1(&i).EOCF_DS_AL2_WRK.EOCF_GOTO_BTN.Enabled = True;
      Else
         &objTermDefn = &objDlFactory.getTermDefn(&termId, False,
            &obj DLConstants.ALLOCATIONTYPE_READONLY);
         &rs1(&i).EOCF_DS_AL2_WRK.EOCF_GOTO_BTN.Label = &objTermDefn.
            EOCF_TERM_ LABEL;
         &rs1(&i).EOCF_DS_AL2_WRK.EOCF_GOTO_BTN.Enabled = False;
      End-If;
   Else
/*************/
&rs1(&i).EOCF_DS_AL2_WRK.EOCF_GOTO_BTN.Label = " ";
 &rs1(&i).EOCF_DS_AL2_WRK.EOCF_GOTO_BTN.Enabled = False;
   End-If;
End-For;

Local Grid &TERMGRID;
Local GridColumn &TERMCOLUMN, &LOOKUPCOLUMN;
```

 Copyright © 1992-2006, Oracle. All rights reserved.

```
&TERMGRID = GetGrid(Page.EOCF_DSPL_ALRT_CFG, "EOCF_DS_ALT_TRM");
&TERMCOLUMN = &TERMGRID.GetColumn("EOCF_GOTO_BTN");
&TERMCOLUMN.Label = " ";
&LOOKUPCOLUMN = &TERMGRID.GetColumn("EOCF_CONFIGURE");
&LOOKUPCOLUMN.Label = MsgGetText(18112, 2541, "Message not found - Get Term");
/*****************************************************************/
/*POPULATE RULE INFO - NEEDS TO BE DONE BY ALL CLF ACTIONS, i.e.,    */
/*ACTIONS WHOSE CONFIG PAGES HAVE BEEN NAVIGATED TO VIA RULE BUILDER */

&objRuleBuilder.PopulateRuleInfo(&gobjRule);
/*****************************************************************/
/* GET THIS ACTION'S ACTION TYPE NAME - TO BE DONE BY ALL ACTIONS */

&rs_level0 = GetLevel0();
&actionId = &rs_level0(1).EOCF_ACTION_WRK.EOCF_ACTION_ID.Value;
&actionTypeId = &rs_level0(1).EOCF_ACTION_WRK.EOCF_ACTION_TYP_ID.Value;
&actionName = &rs_level0(1).EOCF_ACTION_WRK.EOCF_ACTION_NAME.Value;

/* Get details of Action Type */
rem SQLExec(SQL.EOCF_ACTTYP_DTLS2_SEL, &actionTypeId, &actTypeName,
   &dtAppClassId, &dtAppClsPath, &rtAppClassId, &rtAppClsPath, &isTerminal,
   &isConfigurable, &is Combinable, &aeTriggered, &appMsgTriggered,
   &ciTriggered, &piaTriggered, &Commit Flag, &dtCommit, &descr);

/***** Get details of Action Type ***/
/**** Modified by SB to enable rel. language processing for action type name ***/

&rsActionTypeDefn = CreateRowset(Record.EOCF_ACTN_TYPE);
&numrows = &rsActionTypeDefn.Fill("WHERE EOCF_ACTION_TYP_ID = :1",
   &actionTypeId);

If &numrows > 0 Then
   &actTypeName = &rsActionTypeDefn(1).GetRecord(1).EOCF_ACT_TYP_NAME.Value;
   &dtAppClassId = &rsActionTypeDefn(1).GetRecord(1).EOCF_DT_APPCLASSID.Value;
   &dtAppClsPath = &rsActionTypeDefn(1).GetRecord(1).EOCF_DT_APPCLSPATH.Value;
   &rtAppClassId = &rsActionTypeDefn(1).GetRecord(1).EOCF_RT_APPCLASSID.Value;
   &rtAppClsPath = &rsActionTypeDefn(1).GetRecord(1).EOCF_RT_APPCLSPATH.Value;
   &isTerminal = &rsActionTypeDefn(1).GetRecord(1).EOCF_IS_TERMINAL.Value;
   &isConfigurable = &rsActionTypeDefn(1).GetRecord(1).EOCF_IS_CFG_FLAG.Value;
   &isCombinable = &rsActionTypeDefn(1).GetRecord(1).EOCF_IS_CMBIN_FLAG.Value;
   &aeTriggered = &rsActionTypeDefn(1).GetRecord(1).EOCF_AE_TRIGGERED.Value;
   &appMsgTriggered = &rsActionTypeDefn(1).GetRecord(1).EOCF_APPMSG_TRIGGR.Value;
   &ciTriggered = &rsActionTypeDefn(1).GetRecord(1).EOCF_CI_TRIGGERED.Value;
   &piaTriggered = &rsActionTypeDefn(1).GetRecord(1).EOCF_PIA_TRIGGERED.Value;
   &CommitFlag = &rsActionTypeDefn(1).GetRecord(1).EOCF_COMMIT_FLAG.Value;
   &dtCommit = &rsActionTypeDefn(1).GetRecord(1).EOCF_DT_COMMIT.Value;
   &descr = &rsActionTypeDefn(1).GetRecord(1).DESCR.Value;
End-If;
```

Copyright © 1992-2006, Oracle. All rights reserved. 

```
/* Populate the ActionTypeName field for display */
&rs_level0(1).EOCF_DSTIME_WRK.EOCF_ACT_TYP_NAME.Value = &actTypeName;

/*****************************************************************/
/* PLEASE DO THE FOLOWING IF YOU NEED THE DETAILS OF THE ACTION TYPE */
/* THAT WERE ENTERED AT THE TIME OF ACTION TYPE Registration*/
/*  This Display Alert action Does not need these details*/
/*
&rs_level0(1).EOCF_DSTIME_WRK.EOCF_DT_APPCLASSID.Value = &dtAppClassId;
&rs_level0(1).EOCF_DSTIME_WRK.EOCF_DT_APPCLSPATH.Value = &dtAppClsPath;
&rs_level0(1).EOCF_DSTIME_WRK.EOCF_RT_APPCLASSID.Value = &rtAppClassId;
&rs_level0(1).EOCF_DSTIME_WRK.EOCF_RT_APPCLSPATH.Value = &rtAppClsPath;
&rs_level0(1).EOCF_DSTIME_WRK.EOCF_IS_CFG_FLAG.Value = &isConfigurable;
&rs_level0(1).EOCF_DSTIME_WRK.EOCF_IS_CMBIN_FLAG.Value = &isCombinable;
&rs_level0(1).EOCF_DSTIME_WRK.EOCF_IS_TERMINAL.Value = &isTerminal;
&rs_level0(1).EOCF_DSTIME_WRK.EOCF_APPMSG_TRIGGR.Value = &appMsgTriggered;
&rs_level0(1).EOCF_DSTIME_WRK.EOCF_CI_TRIGGERED.Value = &ciTriggered;
&rs_level0(1).EOCF_DSTIME_WRK.EOCF_AE_TRIGGERED.Value = &aeTriggered;
&rs_level0(1).EOCF_DSTIME_WRK.EOCF_PIA_TRIGGERED.Value = &piaTriggered;
&rs_level0(1).EOCF_DSTIME_WRK.EOCF_COMMIT_FLAG.Value = &CommitFlag;
&rs_level0(1).EOCF_DSTIME_WRK.EOCF_DT_COMMIT.Value = &dtCommit;
&rs_level0(1).EOCF_DSTIME_WRK.DESCR.Value = &descr;
*/
/*****************************************************************/
/* PLEASE DO THE FOLLOWING IF  THE ACTION IS A COMBINABLE ONE AND YOU */
/* NEED THE INFO REGARDING THE ACTION BUNDLE OF WHICH IT IS A PART   */
/* This Display Alert Action , though combinable, does not need this */
/*
SQLExec(SQL.EOCF_ACT_BUNDLE_SEL, &actionTypeId, &ActnBundleId);
&rs_level0(1).EOCF_DSTIME_WRK.EOCF_ACT_BUNDLE_ID.Value = &ActnBundleId;*/
/*****************************************************************/
```

# Creating Design and Runtime Application Class Code

To create the design-time and runtime application class code:

1. Create an application package named <*your product code*>*_AAF_ACTIONS*.
2. Create a class in the package to be your design-time action application class.

   It must extend the EOCF_CLF_AF:ActionCfgBase and must implement the designAction() method. This class is responsible for transferring from the policy builder page to the action configuration page previously created.

   All relevant information regarding the rule and the context will be available to the configuration page.

   (See the EOCF_CLF_ACTIONS:DisplayAlertCfg as an example.)
3. Create another class in the package to be your runtime application class.

   It must extend EOCF_CLF_AF:ActionBase and must implement the runAction() method. This class is responsible for launching the action.

   All the relevant information that the action needs to perform will be available to it.

 Copyright © 1992-2006, Oracle. All rights reserved.

(See the EOCF_CLF_ACTIONS:DisplayAlert as an example.)

4. Register the action type in the Register Action Type page.

   Specify the name of your new action type, the names of your design-time and runtime application classes and other action type characteristics .

   Specify the trigger types and trigger points that may have policies using this action type. For example you may want an action to be triggered during ComponentPostBuild trigger types, specifically the trigger point “When a Defect is Presented, in Quality Management.” After completing these configuration steps, the action type will be available for use.

5. If an action type is combinable with other action types, register it in an action type bundle in addition to the other combinable action types.

   All display action types must be included in a single action type bundle.

6. If your new action is configurable, click Configure on the page.

   This transfers you to the action configuration page for more specifications.

7. Insert the subpage EOCF_DISPLAY_SUBPG in Level 0 on all pages that you want display-enabled.

**Note.** This display subpage must not overlap any other field and there should be no popup blockers running.

# Considerations When Creating an Application Class Implementation

PeopleSoft recommends:

- Implementations of related terms be grouped in a single class in which different methods are responsible for deriving the term’s value. This practice facilitates maintaining a manageable number of application classes.
- In situations where the probability that multiple related terms are accessed during a single business event is high:
  - It is preferable to have a single implementation return a rowset, record, or vector containing the data for as many of the terms as possible.
  - When defining the term, specify which data element in the rowset, record, or vector is to be used for the term.

## Technical Details of Application Class Implementations

The data library calls the appropriate application class based on the class and package information registered with the implementation. The application class is instantiated automatically and the specified application class method is invoked. The application class constructor must be coded to accept an instance of the AccessMethod class.

For example:

```
class LeadOppMetrics
 method LeadOppMetrics(&_oAccessMethod As EOCF_CLF_DL:Runtime:AccessMethods:
   Access Method);
 method LeadCountbyBO();
    private instance EOCF_CLF_DL:Runtime:AccessMethods:AccessMethod
```

Copyright © 1992-2006, Oracle. All rights reserved. 

```
        &ioAccess Method;
end-class;
```

The AccessMethod object allows the application class to access values for all the implementation binds (input parameters) needed by the implementation and all the information about the term being resolved.

## Example of an Application Class Accessing Implementation Binds

The following partial code shows how an application class can access any of its implementation binds.

```
method GetCaseID Local any &CaseID;
  &CaseID = &ioAccessMethod.getBindValueByName("CASE_ID");
  Local EOCF_CLF_DL:Runtime:Results:ResultsScalar &oResultsScalar;
  &oResultsScalar = create EOCF_CLF_DL:Runtime:Results:ResultsScalar(&CaseID);
      &ioAccessMethod.Results = &oResultsScalar;
end-method;
```

The application class does not need to know how the data is retrieved and passed by the data library engine. The data could have been passed by the calling application; could have been defined as a constant in context definition; or, could have been defined as a term and resolved by the data library engine. The application class can retrieve the data either by position or by name.

```
/* BO_ID is the name of the implementation bind;the bind name specified
 here matches the one to be specified at the time of registering the
 implemenation in &nBOID = &ioAccessMethod.getBindValueByName("BO_ID");
 Here the request is made to retieve the value for the first bind. */
&nBOID = &ioAccessMethod.getBindValueByPosition(1);
```

Another way the calling application can pass additional information to implementations or policy actions is to have the calling application use the addUserVariable method to add the information to the context object. To access this information, the application class uses the getUserVariable method to retrieve the content.

For example:

```
&AdditionalData = &oAccessMethod.AMContext.getUserVariable("ExtraInfo);
```

---

**Note.** The shape of the data received by an application class using any of the previous methods will be a PeopleCode Any object. Therefore, it's the responsibility of the application class to convert the object to the appropriate data type before the value is consumed. If the application class has a method to be invoked by the data library engine, then that method should not require any parameters.

---

## Sample Methods of an Application Class Resolving Terms

The following code is an example of an application class having several methods for resolving terms.

```
class OperatorInfo
 method Get_Person_Name();
  method Get_Person_Salutation();
 method Get_Person_Title();
 method Get_Person_Gender();
 method Get_Person_BirthDate();
  method OperatorInfo(&_oAccessMethodParam As EOCF_CLF_DL:Runtime:
     AccessMethods: AccessMethod);
  private
```

 Copyright © 1992-2006, Oracle. All rights reserved.

```
   instance EOCF_CLF_DL:Runtime:AccessMethods:AccessMethod &ioAccessMethod;
end-class;
```

If the data library engine invokes an application class method, that method is responsible for deriving the results and for populating the results in the AccessMethod object. The data library engine transmits the results to the calling application. If an application class method is not specified in the implementation definition, the constructor is responsible for performing these tasks.

The data library provides several mechanisms through which the application class can return the output value to the data library:

- The application class instantiates one of the following application classes with the output value that needs to be passed to the engine.
  - `ResultsRecord Record &oResultsRecord = create EOCF_CLF_DL:Runtime: Results:ResultsRecord(RecordVariable);`
  - `ResultsRowset Rowset &oResultsRowset = create EOCF_CLF_DL:Runtime: Results:ResultsRowset(RowsetVariable);`
  - `ResultsScalar Scalar &oResultsScalar = create EOCF_CLF_DL:Runtime: Results:ResultsScalar(ScalarVariable);`
  - `ResultsVector Vector &oResultsVector = create EOCF_CLF_DL:Runtime: Results:ResultsVector(VectorVariable);`
- The type of class to be instantiated depends upon the type of data that needs to be returned.
- The instantiated object is assigned to the member of AccessMethod object.

---

**Note.** Application class objects are not cached. When the data library engine invokes the application class because the data is not available in the cache, the application class is instantiated by invoking the constructor, then the method specified in the implementation. Therefore, the instance variables created in the constructor cannot be shared across multiple methods of the same class to resolve different terms.

---

See Chapter 10, "Setting Up the Data Library," Understanding Term Properties, page 130.

## Example of How a Value Can Be Passed to the Data Library

The following sample is a partial code listing of how a value can be passed to the data library:

```
method GetCaseID
Local any &CaseID;
   &CaseID = &ioAccessMethod.getBindValueByName("CASE_ID");
/* use the appropriate sub class of Results class for assigning value */
Local EOCF_CLF_DL:Runtime:Results:ResultsScalar &oResultsScalar;
&oResultsScalar = create EOCF_CLF_DL:Runtime:Results:ResultsScalar(&CaseID);
&ioAccessMethod.Results = &oResultsScalar;end-method;
```

## Sample Code of an Application Class Implementation

The following is a sample of an application class implementation.

```
import EOCF_CLF_DL:Runtime:AccessMethods:*;
import EOCF_CLF_DL:Runtime:Resolution:*;
import EOCF_CLF_DL:Contexts:*;
```

Copyright © 1992-2006, Oracle. All rights reserved. 

```
import EOCF_CLF_DL:Runtime:Results:*;

class CaseRecordInformation
   method CaseRecordInformation(&_oAccessMethod As EOCF_CLF_DL:Runtime:
      Access Methods:AccessMethod);
   method GetResultRecord();

private
   instance EOCF_CLF_DL:Runtime:AccessMethods:AccessMethod &ioAccessMethod;
end-class;

method CaseRecordInformation
   /+ &_oAccessMethod as EOCF_CLF_DL:Runtime:AccessMethods:AccessMethod +/

   &ioAccessMethod = &_oAccessMethod;
   end-method;

method GetResultRecord
   Local Record &result;
   Local number &nResolved_value, &nCaseID;
   Local string &sBusUnit;

/* Retrieving the implementation bind using the API */
   &nCaseID = &ioAccessMethod.getBindValueByName("CASE_ID");

   SQLExec("SELECT BUSINESS_UNIT FROM PS_RC_CASE WHERE CASE_ID = :1",
      &nCaseID, &s BusUnit);
   &result = CreateRecord(Record.RC_CASE);
   &result.CASE_ID.Value = &nCaseID;
   &result.BUSINESS_UNIT.Value = &sBusUnit;
      &result.SelectByKey();
/* Passing the values (in this case, it is a record) back to the data
   library engine - this record could be used to resolve multiple terms */
   &ioAccessMethod.Results = (create EOCF_CLF_DL:Runtime:Results:
      ResultsRecord (&result));

end-method;
```

# Considerations When Creating a PS Query Implementation

PeopleSoft recommends that when multiple related terms may be accessed during a single business event:

- Create a single PS Query implementation to return a rowset containing the data for several terms
- Specify which data element or field position in the rowset or record is to be used for the term.
- Ensure appropriate privileges have been set for accessing the objects present in the query.

 Copyright © 1992-2006, Oracle. All rights reserved.

**Note.** Use caution when using this type of implementation because the query may execute very slowly.

# Considerations When Creating a Policy

PeopleSoft Active Analytics Framework is neither a programming tool nor a substitute for PeopleCode. Do not use the framework as a way to program policies that do not need to be configurable. If policies are static, the use of PeopleCode is the best method. The framework should not be used as a tool to configure the presentation layer: for dynamically hiding or unhiding fields, making the fields editable or not, and so on. Furthermore, the data library should not be used when there is no need for the data abstraction layer.

Use the PeopleSoft Active Analytics Framework when there's a need for business users to configure business processes with business rules, without having to customize the application.

Before building policies, consider that each term used in conditions or in actions within policies, could negatively affect performance of the application component, especially if the term's retrieval mechanism is time consuming.

Copyright © 1992-2006, Oracle. All rights reserved.

Copyright © 1992-2006, Oracle. All rights reserved.

# PART 5

# Enterprise Integration

**Chapter 15**
**Understanding Enterprise Integration**

**Chapter 16**
**Understanding Integration Points**

**Chapter 17**
**Activating Integration Points**

**Chapter 18**
**Assigning Publishing Rules**

**Chapter 19**
**Using the Error Handling Utility**

**Chapter 20**
**Using the Effective Date Publish Utility**

**Chapter 21**
**Using the Flat File Utility**

**Chapter 22**
**Using the XML Schema Utility**

# CHAPTER 15

# Understanding Enterprise Integration

This chapter gives an overview of PeopleSoft's Enterprise Integration and discusses:

- PeopleSoft Business Interlinks.
- PeopleSoft Component Interfaces.
- File layouts.
- PeopleSoft Integration Broker.

## Understanding PeopleSoft Business Interlinks

PeopleSoft Business Interlinks enables you to perform component-based, real-time integration from PeopleSoft to external systems. PeopleSoft Business Interlinks creates synchronous transactions that enable PeopleSoft applications to pass data to and receive data from the external system in real time. You can use PeopleSoft Business Interlinks to integrate PeopleSoft with third-party systems, with another PeopleSoft application, or with systems on the internet.

## Understanding PeopleSoft Component Interfaces

A component interface is a set of application programming interfaces (APIs) that you can use to access and modify PeopleSoft database information using a program instead of the PeopleSoft client. PeopleSoft Component Interfaces expose a PeopleSoft component (a set of pages grouped for a business purpose) for synchronous access from another application (PeopleCode, Java, C/C++, or Component Object Model [COM]). A PeopleCode program or an external program (Java, C/C++, or COM) can view, enter, manipulate, and access PeopleSoft component data, business logic, and functionality.

## Understanding File Layouts

A file layout is a definition (or mapping) of a file to be processed. It identifies where fields are located in file data. Once you create a file layout, you can write PeopleCode programs that ultimately use file layout, either to read data from or write data to a file.

In addition to manipulating transaction data, you can use file layouts and flat files to move data between your PeopleSoft database and external systems (data interchange).

Copyright © 1992-2006, Oracle. All rights reserved.

# Understanding PeopleSoft Integration Broker

PeopleSoft Integration Broker is a messaging system that enables you to synchronize data from one application or system with another.

PeopleSoft Integration Broker facilitates synchronous and asynchronous messaging among internal systems and trading partners, while managing message structure, message format, and transport disparities.

PeopleSoft Integration Broker comprises two high-level subsystems: the integration engine and the integration gateway. The integration engine runs on the PeopleSoft application server. It is tied closely to PeopleSoft applications and produces or consumes messages for these applications.

The integration gateway manages the receipt and delivery of messages passed among systems through PeopleSoft Integration Broker. It provides support for the leading TCP/IP protocols used in the marketplace today, and more importantly, provides extensible interfaces for the development of new connectors for communication with legacy and internet-based systems.


Copyright © 1992-2006, Oracle. All rights reserved.

# CHAPTER 16

# Understanding Integration Points

This chapter discusses integration points.

**Note.** The term *integration point* is synonymous with the term *Enterprise Integration Point (EIP)* and may appear interchangeably in this documentation.

## Overview of Integration Points

Provided by a PeopleSoft application, an integration point is an interface that is used to communicate with other PeopleSoft or external applications. An integration point provides integration details for PeopleSoft applications, including which technologies are involved, technology details, information for using PeopleSoft Integration Broker for messaging, and how different integration points are related.

The integration point consists of data rules for the applications that it supports. The integration points that are delivered with PeopleSoft provide generic functionality so that they can be adapted for use with as many programs as possible.

An integration point can be implemented by using different technologies available in PeopleTools, such as messaging, component interfaces, business interlinks, XML links, and electronic data interchange (EDI).

Integration points can be associated with or used by application groups. An application group is a logical grouping of applications that use an integration point in the same business manner.

Other than this grouping facility, an application group and an application mean the same thing. In the rest of the documentation, the words *application group* and *application* are used interchangeably unless clearly specified.

Every integration point is owned by at least one application, but can be used in multiple applications. Therefore, if an application sends an integration point, and another application can receive the same integration point, the two systems should be interoperable, assuming the data structure and the rules of the integration point are implemented the same in both places.

However, sometimes two applications might use the same integration point but implement it in different ways. For example, one application that uses the Customer integration point may need to transform the data before it can be sent to an external system, which has another data structure for its customer information.

An integration point can be a part of multiple application groups. For example, the Department Table integration point may be used by a number of application groups, including PeopleSoft Human Resources, CRM, and General Ledger.

See Interactive Services Repository on the PeopleSoft Customer Connection website.

Copyright © 1992-2006, Oracle. All rights reserved.

 Copyright © 1992-2006, Oracle. All rights reserved.

# CHAPTER 17

# Activating Integration Points

This chapter discusses how to:

- Set up PeopleSoft integration points.
- Set up related languages.
- Examine related-language messaging scenarios.

## Setting Up PeopleSoft Integration Points

This section discusses how to:

- Set up publication rules.
- (Optional) Map nodes to a chunking rule.
- (Optional) Assign business units to a chunk rule.
- (Optional) Assign setIDs to a chunking rule.
- Specify OnRoute PeopleCode.

**Note.** For more information and technical details about these integration points, and for information on how and with what applications to use them, consult the relevant application PeopleBook.

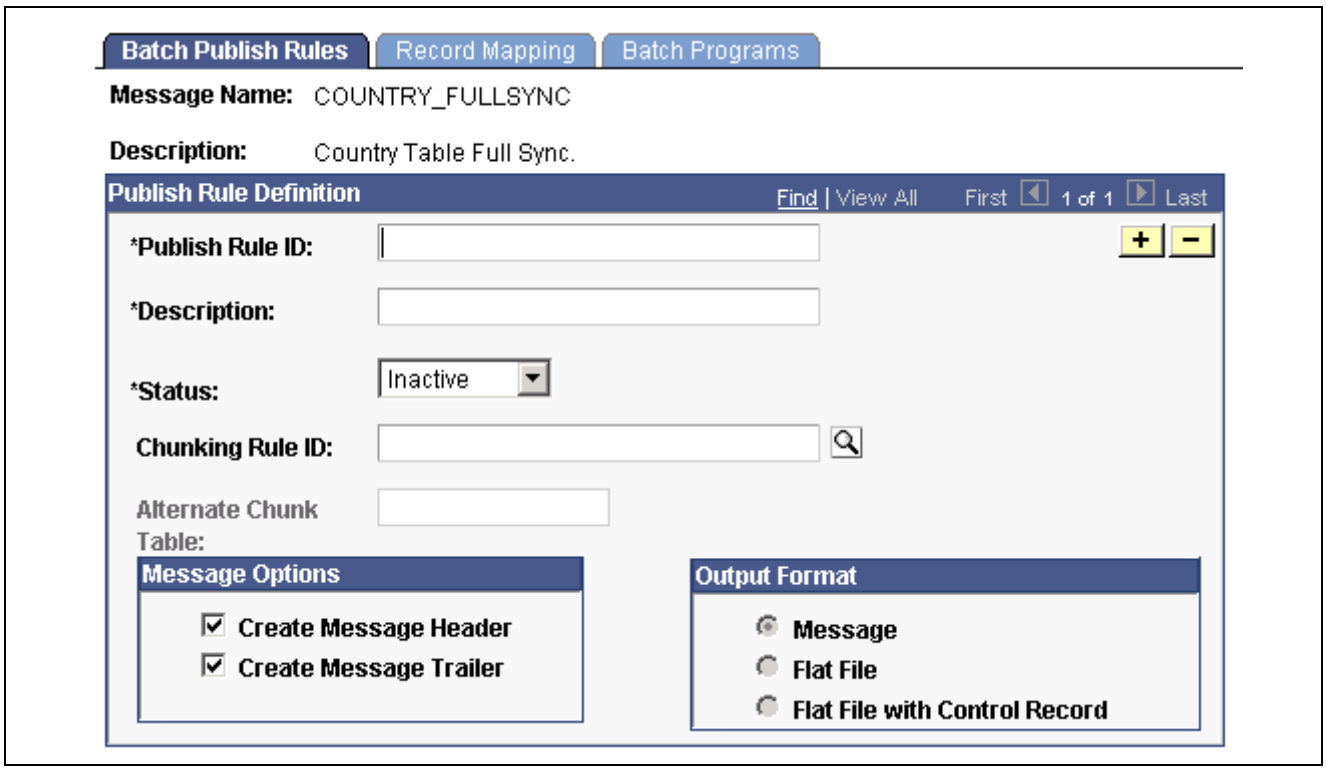
### Pages Used to Set Up PeopleSoft Integration Points

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Batch Publish Rules | EO_MSGPUBATCH | Enterprise Components, Integration Definitions, Batch Publish Rules | Set up publication rules.<br>You must activate a publication rule for the publication messages that you create to follow. This rule includes instructions on message chunking, if necessary. |
| Add Nodes to Chunk Rule | EO_ADNODECHUNK_PNL | Enterprise Components, Integration Definitions, Map Chunking Rules, Node to ChunkRule | Map PeopleSoft message nodes to chunking rules. |

Copyright © 1992-2006, Oracle. All rights reserved.

# Setting Up Publication Rules

Access the Batch Publish Rules page.




Batch Publish Rules page

If the data that you're transmitting does not fit in a single message, or if you want to send different parts of the message to different target systems, set up the rules to chunk the message and associate it with the publish rule. The business unit and setID chunking rules are standard in PeopleSoft applications, but you can configure chunking rules.

**Publish Rule ID** Select the name of the message for which you're setting up rules.

**Status** Select *Active* to activate this publish rule definition for this message. Select *Inactive* to prevent this rule from applying to this message.

**Chunking Rule ID** and **Alternate Chunk Table** Enter the unique chunking rule name that is set up when you created the chunking rule. The message that you publish is routed based on this field. If necessary, enter an additional field in the Alternate Chunk Rule ID field by which to chunk the message.

## Message Options

Many PeopleSoft systems rely on a message header and message trailer to trigger subscription PeopleCode to discard old table data and insert the new incoming data. As a general rule, all FullSync messages should use a header and trailer. Sync messages don't need headers and trailers.

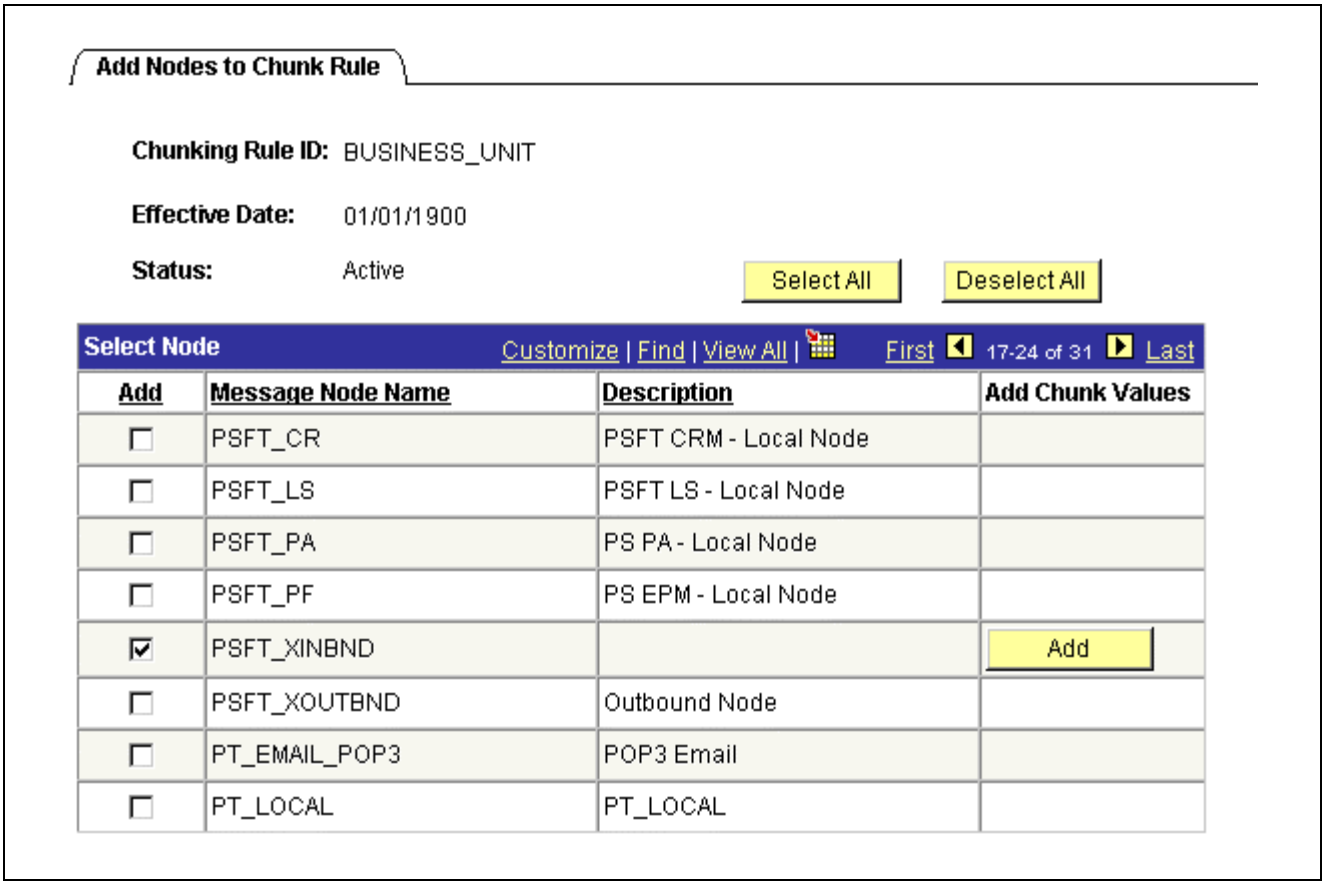
## Output Format

The Application Engine program used to chunk messages can create either an XML message that flows through messaging architecture or a flat file that is generated in PeopleSoft Process Scheduler and not published elsewhere. Always select Message as the format when you send data to PeopleSoft systems.


Copyright © 1992-2006, Oracle. All rights reserved.

## Mapping Nodes to a Chunking Rule

Access the Add Nodes to Chunk Rule page.




Add Nodes to Chunk Rule page

To map nodes to a chunk rule:

1. In the Add column, select the check box next to the nodes that you defined earlier.

   After you select a node, use the Add button in the Add Chunk Values column to open the Quick Map page for the message you defined earlier.

2. Click Save.

## Assigning Business Units or SetIDs to a Chunking Rule

See Chapter 18, "Assigning Publishing Rules," Assigning Business Units or SetIDs to a Chunking Rule, page 208.

## Specifying OnRoute PeopleCode

Create a service operation handler to route the message chunk to the correct subscriber node.

To do so, extend the IRouter application class and use the OnRouteSend method. Then in the appropriate service operation, define a handler and specify the application class.

Copyright © 1992-2006, Oracle. All rights reserved.

### See Also

*Enterprise PeopleTools 8.48 PeopleBook: PeopleSoft Integration Broker,*"Sending and Receiving Messages," Understanding Integration PeopleCode

*Enterprise PeopleTools 8.48 PeopleBook: PeopleSoft Integration Broker,*"Sending and Receiving Messages," Messaging Handlers

*Enterprise PeopleTools 8.48 PeopleBook: PeopleSoft Integration Broker,* "Managing Service Operations," Configuring Service Operation Definitions

# Setting Up Related Languages

This section provides an overview of related language tables and related language guidelines for PeopleSoft messaging and discusses how to:

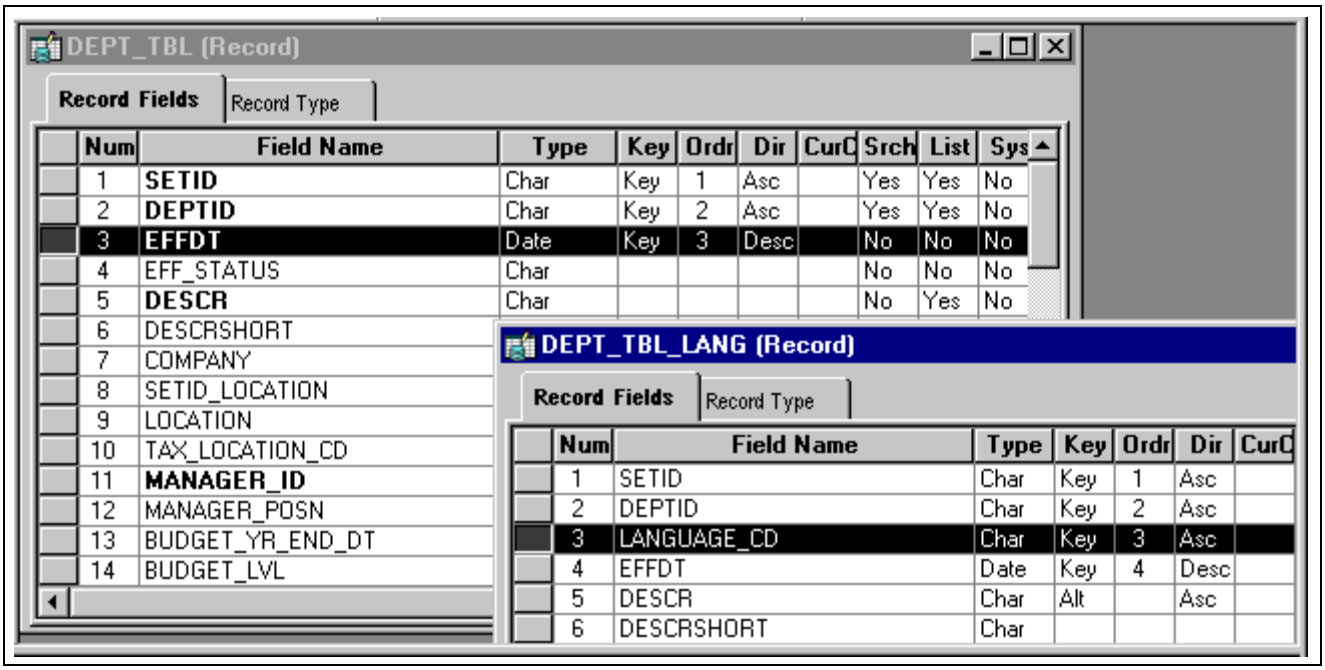
- Interpret the component processor behavior.
- Publish a service operation from a component.
- Publish a service operation from batch programs.
- Subscribe to data in a PeopleSoft multilingual environment.
- Subscribe to data in a non-multilingual environment.

## Understanding Related Language Tables

There are several possible scenarios that you can use to familiarize yourself when setting up related languages for a message.

A department table, for example, must publish information in German as well as English. In the following screen shot, the base application tables and related-language tables have a parent-child relationship. The related-language table has the same name as the parent table, but is suffixed with _LANG, in accordance with PeopleSoft naming conventions.

 Copyright © 1992-2006, Oracle. All rights reserved.



Related-language record definitions

Consider the following:

- The parent table, DEPT_TBL, has the related-language child table, DEPT_TBL_LANG.
- The child table has the same fields as the parent table, plus an additional field of LANGUAGE_CD.
- The child table's attributes are all of the translatable textual fields of the parent record.

## Understanding Related Language Guidelines for Messaging

When publishing a full message, generate messages that contain the contents of an entire table by first generating a message in the base language of the system that contains the full table contents. Then generate messages for each of the related languages that are supported by the publishing system. Each message should contain the full message structure for that message object (levels 0, 1, 2, and so on). The language-specific messages should contain the translatable field values for that language and include the base language fields that are not translatable.

When subscribing to a full message, specify the language code only at level 0 of the message. This captures and sets the user's preferred language to level 0 of the PeopleSoft Common Application Message Attributes (PSCAMA) message header. All data within the message must be in the same language. Follow these steps:

1. Delete the base language tables and related-language tables.
2. Replace these tables with data from the messages as appropriate.
3. Place only those related-language field values that are supported by the subscribing system into the related-language tables.
4. Add the related-language table entry only if the base language table entry already exists.

When publishing an incremental message, the PeopleSoft system generates base messages in the user's preferred language by using the user ID's language code. Putting the user's preferred language code in the message header PSCAMA record defines the message language for the subscribing system.

Copyright © 1992-2006, Oracle. All rights reserved. 

When subscribing to an incremental message using PeopleSoft Component Interfaces, use a simple PeopleCode program that performs a SetLanguage (messagelanguage) call to a component interface with the message definition. This enables the subscribing system to process the data in the appropriate related or base language for that system.

When subscribing to an incremental message using PeopleCode only, the PeopleCode program must simulate what the component processor does. The PeopleSoft system provides a generic Subscribe_IncrReplication PeopleCode function that provides basic language-related ability for incremental message subscriptions.

---

**Note.** All PeopleSoft subscription processes that are associated with textual information work as if the content is related-language enabled; thus the processes provide support for customer related-language extensions and future PeopleSoft enhancements.

For PeopleSoft-to-PeopleSoft system integration, you do not need to specify the language-sensitive data on either system.

---

All of the PeopleCode functions that are needed for related language processing of incremental and full messages are in the FUNCLIB_EOEIP record. The record contains two functions:

- Subscribe_IncrReplication has related-language processing for an incremental message subscription process.
- Subscribe_FullReplication has related-language processing for a batch subscribe process.

## Interpreting Component Processor Behavior

When you open a component, the component processor:

1. Compares the user's preferred language against the base language for the database.
2. Uses the record information from the base application table (DEPARTMENT_TBL).

   If a record in the base application table exists for the user's preferred language, the fields on the related-language table (DEPARTMENT_LANG) overlay the record information. For example, a German user sees German descriptions even if the base language for the database is English.

When you change the user's preferred language and save the component:

1. The component processor writes all the data for related-language fields back to the related-language table.
2. The component processor writes the rest of the data back to the base application table.
3. The German user's entries for the DESCR and DESCRSHORT fields are saved back to the DEPARTMENT_LANG table with its key values and the LANGUAGE_CD field in German.
4. The data that was entered by the German user in the key fields, as well as MANAGER_NAME and ACCOUNTING_OWNER fields, are saved on the parent record DEPARTMENT_TBL.

## Publishing a Message from a Component

The PeopleSoft system employs the user's preferred language to determine the language of a message that is published from a component. The default for LANGUAGE_CD is set to the preferred language code (OPERATOR.LANGUAGE_CD). The standard for incremental changes is to publish only the data that has changed, in the language to which it was changed. Changing the preferred language to translate data generates new messages appropriately.

 Copyright © 1992-2006, Oracle. All rights reserved.

# Publishing a Message from Batch Programs

Application Engine and Structured Query Report incremental message programs should use the base language of the system. These programs perform their accesses and updates on the base tables only, even if related-language tables are supported for those business objects. Related-language tables are featured in batch program processing only in generating warnings or errors that use the message catalog.

When a batch application program runs, the processing is done in the base language of the system, and messages are generated in only the base language.

# Subscribing to Data in a PeopleSoft Multilingual Environment

The subscription process sets the language for processing the message to the language needed by the subscriber system. For example, if the subscribing system's base language is French, and the PeopleSoft system is sending German data, the subscription process must store the German data in a related-language table and the nontranslatable data in the French base application tables. Neither system's base language matters; only the base language of the subscribing system and the actual message language are used.

# Subscribing to Data in a Non-Multilingual Environment

When handling subscribing systems that do not support multiple languages, you can subscribe to data in these ways:

- Setting specific message publish routing PeopleCode.
- Sending service operations to appropriate nodes.
- Permitting the subscription to process itself.
- Subscribing to data that is specific to an external system.

## Setting Message Publish Routing PeopleCode

Set message publish routing PeopleCode to send only messages in a particular language code to a subscribing node.

The subscribing node does not need to check the language in which the message was generated; any message that it receives is in its language is automatically used to update the subscribing system's database. To implement this option:

- Determine the languages in which messages are published.
- Determine the message language that a subscribing node receives.
- Add PeopleCode routing logic on the publish side to check the language code of the first occurrence in the message record and return a list of subscribing nodes that should receive the message for that language.

The publish routing PeopleCode guarantees that the message is sent to the correct subscribing nodes by using the message language code.

## Sending Service Operations to All Appropriate Nodes

Send service operations to all appropriate nodes regardless of the language and have the subscription routing PeopleCode filter out messages in different languages.

PeopleCode on the subscription routing checks the language code of the first occurrence in the message and controls whether the node should receive the message. To implement this option:

- Ensure that service operations in all languages are sent to all appropriate nodes.

Copyright © 1992-2006, Oracle. All rights reserved. 

- Add the PeopleCode to compare the message language against a hard-coded language value for the subscribing system.

The advantage of putting the logic within the subscription routing PeopleCode is that every message is checked for a language value match.

### Permitting the Subscription to Process Itself

You can permit the subscription process itself, rather than the routing PeopleCode for the service operation and queue, to determine whether the message should be processed for the subscribing system.

The subscription process checks the language code of the PSCAMA record for the first instance of the message against a hard-coded value for the subscribing system. If the language code does not match, the message is ignored. If the message language code does match, it's considered a base language message, and it replaces all data on the subscribing system according to the audit action flags on the message records.

---

**Note.** Generic subscription processes should not filter messages based on language code, to avoid data integrity issues.

---

### Subscribing to Data That Is Specific to an External System

Subscribe to data that is specific to an external system for language code, business unit, or setID requirements that are specific to an external system.

Use the chunking rule and the routing control tables that PeopleSoft supplies to select a portion of the data and send it to a specific node.

Use the PeopleSoft-supplied Publish Header (PublishHdr) component to enter the partitioning views and fields for a message. This chunks the message so that all contents within a single message contain the same partitioning value (such as business unit, setID, or application-specific fields).

Set up the routing control for the message so the message is sent only to the appropriate nodes. The PeopleSoft system supplies a business unit routing control (BU Routing Control) component and a setID routing control component that enable applications to specify for each message which nodes should receive the partitioned message data.

After you set up the chunking rule and routing rules, both the full data publish and batch publish programs partition the data according to the appropriate value and route it accordingly. You can now publish and subscribe to the message.

## Examining Related-Language Messaging Scenarios

Actual related-language messaging scenarios include publishing a non-base language message and switching a preferred language.

This section discusses how to:

- Publish a non-base language message to a PeopleSoft subscribing system with a different base language and no prior data.
- Switch the preferred language to Japanese and update the same employee's name and address.

 Copyright © 1992-2006, Oracle. All rights reserved.

## Publishing a Non-Base Language Message to a Subscribing System With a Different Base Language and No Prior Data

In the following example, the publishing system base language is English, the subscriber base language is Japanese, and the user's preferred language is German.

To publish a non-base language message:

1. An online user adds a new level 0 key on a page.

   The data is stored in both the base language table (English) and the related-language table (German).

2. The system publishes a message in the user's preferred language (German).

3. The data is inserted into the subscribing system.

   The data is inserted into both the base language table and related-language table (German) because it is added for the first time; data cannot reside in a related-language table without corresponding data in the base table.

## Switching the Preferred Language to Japanese and Updating the Same Employee's Name and Address

To switch a preferred language:

1. A user switches the preferred language from English to Japanese.

   The user updates the same record as in the previous scenario. Data that is not language-sensitive is updated in the base table (English). Language-sensitive data is inserted into the related-language table (Japanese).

2. A system publishes a message in the user's preferred language (Japanese).

3. Data is inserted into the subscribing system.

All data goes to the base table (Japanese), because the message was sent in the same language as the subscribing system's base language.

Copyright © 1992-2006, Oracle. All rights reserved. 

 Copyright © 1992-2006, Oracle. All rights reserved.

# CHAPTER 18

# Assigning Publishing Rules

This chapter provides an overview of message publishing rules and discusses how to:

- Assign full table publishing rules.
- Assign batch publishing rules.
- Set up message chunking.

## Understanding Publishing Rules

This section discusses:

- The Publish utility.
- Terms used when defining rules.

### The Publish Utility

The Publish utility automates the process of copying the contents of an entire table into a remote database or legacy system. Use the utility to synchronize data from an existing system when a new PeopleSoft system is installed. The data is chunked based on the MaxMessageSize parameter.

Control the size, number, and frequency of these data messages by using a series of data publishing rules and:

- Whether to create header and trailer records.
- Where the data comes from.
- Whether to chunk the message.
- Which related languages to publish.

The publishing rules include pages for full table publishing rules, batch publishing rules, record mapping, languages, and batch programs.

The MaxMessageSize (maximum message size) field in the PeopleSoft Option (PSOPTIONS) table limits the size of the message. Before processing each level zero record, the Publish utility compares the size of the message against the value in the MaxMessageSize field. When the message size exceeds the value in the MaxMessageSize field, the message publishes and a new message starts. You can also specify message chunking in the publish rules for the message, which enables the message to publish when the value of a chunk field changes.

### Terms Used When Defining Publish Rules

When defining publish rules, you need to know the following terms:

Copyright © 1992-2006, Oracle. All rights reserved.

| | |
|---|---|
| **Alternate Chunking Table** | This secondary chunk table is a separate view of an existing chunk table but with one or more field names customized. It enables you to reuse an existing chunk table. For example, if the record EO_BUSUNT_EOC has BUSINESS_UNIT as the chunking field, you can create a view of this table that has BUSINESS_UNIT_IN as the chunking field. |
| **Batch Publish** | This term describes jobs or processes that run independently from their initiating process. A batch process can also run at one or more predetermined times in the future from the initiating request. A batch process is appropriate for publishing incremental changes to data in a batch environment or when processing large volumes. |
| **Chunking** | Chunking refers to the automatic breaking of a message into several smaller messages based on values in fields in the level zero record. Chunking on business unit means that all transactions within the message are for the same business unit value. |
| **Chunking Fields** | Chunking fields are key fields in the level zero record that are used to break the message into parts. |
| **Chunking Rule** | A chunking rule points to the chunking table. Multiple chunking rules can point to the same chunking table. |
| **Chunking Table** | A chunking table is a derived or Structured Query Language (SQL) table that contains the fields by which the message is chunked. SQL chunk tables define the valid values of the chunking fields and the nodes to which the message is published. |
| **Message Header** | The header is data that precedes a message and triggers the subscription process to initialize tables before receiving the data messages. |
| **Message Trailer** | The trailer is data that follows a message and triggers the subscription process to indicate that all data messages have been received. |
| **Full Publish** | The full publish process seeds, or initially populates or repopulates, a copy of an entire table into a remote database or legacy system. The entire contents of the table are published to all systems that require a copy of the table. |

# Assigning Full Table Publishing Rules

All PeopleSoft applications use common, centralized tables and pages to define how to publish full table messages. The Publish utility uses full table publish rules to process the data.

This section discusses how to:

- Associate a rule to a message and characterize the rule.
- Map a message record to another record.
- Specify languages in which to publish messages.

**Note.** You can create multiple publish rules for the same message. The Publish utility treats each publish rule as a separate publishing cycle.

 Copyright © 1992-2006, Oracle. All rights reserved.

## Pages Used to Assign Full Table Publishing Rules

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Full Table Publish Rules | EO_MSGPUBFULL | Enterprise Components, Integration Definitions, Full Data Publish Rules | Associate a rule to a message and characterize the rule. |
| Record Mapping | EO_MSGRECMAP | Enterprise Components, Integration Definitions, Full Data Publish Rules, Record Mapping | Map a message record to another record. |
| Languages | EO_MSGLANGUAGE2 | Enterprise Components, Integration Definitions, Full Data Publish Rules, Languages | Specify languages in which to publish a message. |

## Associating a Rule to a Message

Access the Full Table Publish Rules page.




Full Table Publish Rules page

**Publish Rule ID** Enter an ID for the user-defined rule to associate with this message. Assign a logical name to make the rule easy to find.

**Status** Select *Active* to activate the publish definition.

**Create Message Header** and **Create Message Trailer** Ensure that the subscribing process does not need the header or trailer process before you clear these check boxes for a Batch Publish message. Header messages trigger special logic (in a PeopleCode program) on the PeopleSoft full message subscription that deletes the existing application records. Also,

Copyright © 1992-2006, Oracle. All rights reserved. 

some applications use the trailer message to indicate that all data messages have been received and to initiate the validation process. The documentation for the individual message should note whether headers and trailers are supported.

**Create Delay Records** This check box appears only if the message name ends with *FULLSYNC_EFF* (such as MESSAGENAME_FULLSYNC_EFF). Select this check box to write all future-dated rows to the delay table. Also select this check box in conjunction with record mapping views that publish only the current effective-dated rows.

### Output Format

These options indicate the output of the Publish utility. Valid output formats are:

- Message (default value).
- Flat file.
- Flat file with control record.

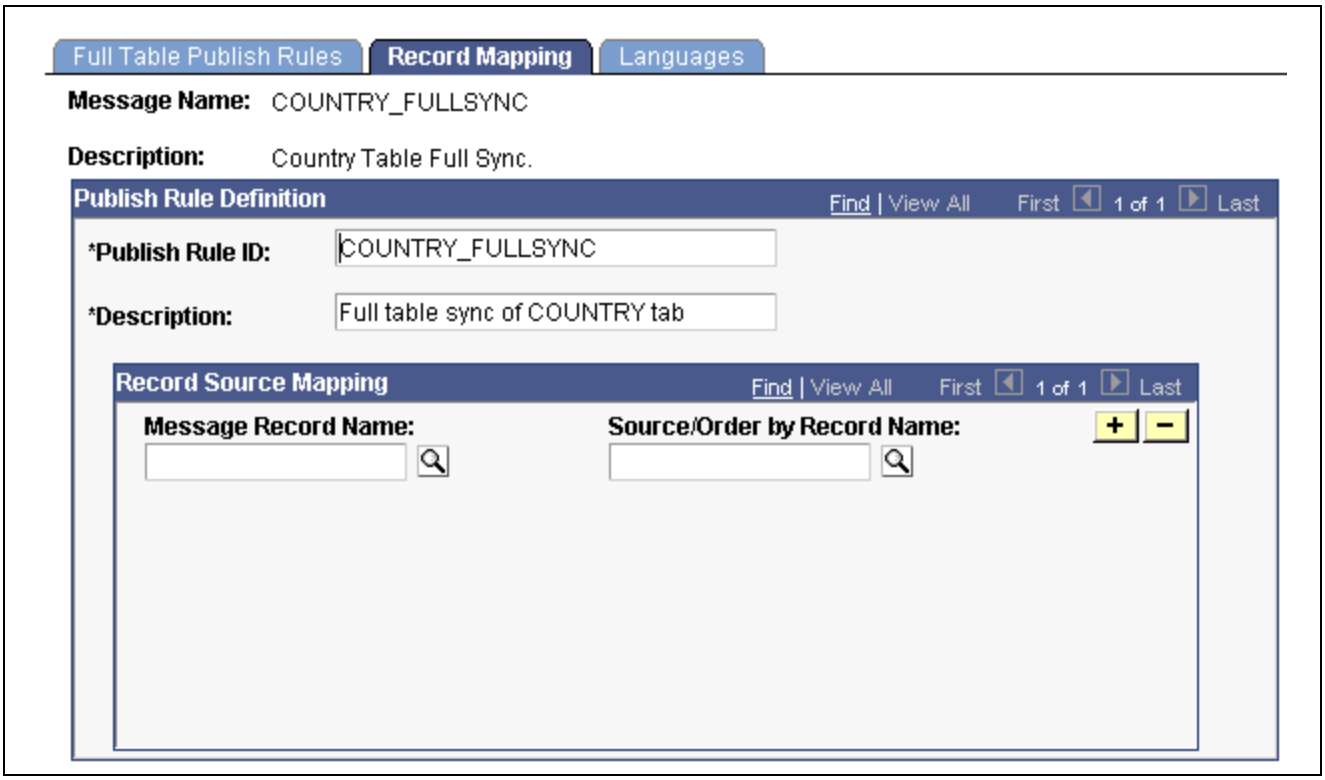
Select the flat file option when you set up publish rule definitions for the message. If there is a file layout object with the same name as the message object, you can modify the output format field on the Publish Rule Definition page. You can create multiple publish rules with different output formats for the same message. If you select flat file output, the header and trailer messages aren't created, and a single output flat file is created even if you already specified the chunking rule. The file layout definition must have a record structure identical to that of the message; if the AUDIT_ACTN field does not exist in the record, you must add it to the file layout record definition.

The option becomes modifiable if there is a file layout object with the same name as the message definition. The directory location of the flat file is determined by the value in the OUTPUT parameter in PeopleSoft Configuration Manager. The flat file name is *messageName_SequenceNumber*.out, where *messageName* is the message name and *SequenceNumber* is the next available sequential number for files with that message name.

## Mapping a Message Record to Another Record

Access the Record Mapping page.

 Copyright © 1992-2006, Oracle. All rights reserved.




Record Mapping page

Specify the source data for a record in a message from:

- A staging table.
- A temporary table.
- A view.

Regardless of which source table that you use, ensure that the source table field names are identical to the field names in the target message record. Key fields must also adhere to the parent and child relationship. (Keys of a parent record must exist in the child record, in the same sequence.) The Publish utility uses the Source/Order By Record Name field to select rows for publishing. You use key matching to find all child rows of a parent.

The chunk field should be the primary key field in all Source/Order By records. If the chunk field is not a key field of the level zero record, join the chunk field to all records in which the chunk field does not exist by using views in PeopleSoft Application Designer.

Three reasons to use the record mapping feature in a publish rule are:

- The published data comes from staging tables or temporary tables.

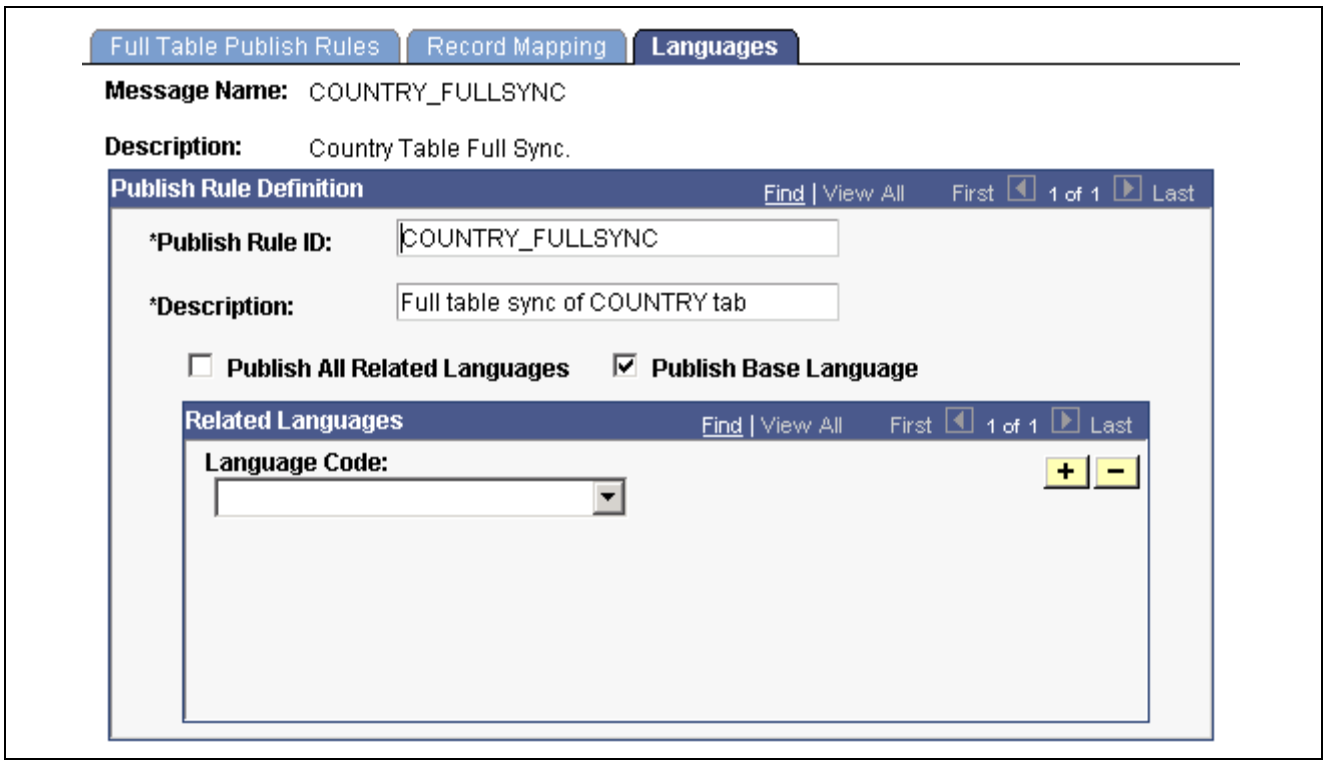
  Using staging tables is the only way to publish rows that have been deleted from application master tables.
- The published data needs an order sequence that differs from that of the records in the message.
- You must create a view that selects only current effective-dated rows in situations where the application table is effective-dated and the subscribing database cannot process future-dated transactions.

Enter only those message records with a different source or ordering record. If the message record name and the source or ordering record name are identical, do not insert a row for that record on the Record Mapping page.

Copyright © 1992-2006, Oracle. All rights reserved. 

# Specifying Languages in Which to Publish Messages

Access the Languages page.




Languages page

**Publish All Related Languages** Select to indicate whether to publish the message in all of the related languages. If selected, the scroll area that you use to enter individual related languages is unavailable. This check box is clear by default.

**Publish Base Language** Select to publish the message in the base language. This check box is selected by default.

**Language Code** Select the related language in which to publish the message.

The Publish utility creates the following related language messages:

- One message in the base language of the publishing system.
- One message for each language in the related language tables for the base tables.

The subscribing system receives the messages in the order in which they are published. For example, if the base language is English, with French and then German as related language tables, the Publish utility creates the messages in this order:

1. (Optional) Header message.
2. English message 1.
3. French message 1.
4. German message 1.
5. English message 2.


Copyright © 1992-2006, Oracle. All rights reserved.

6. French message 2.
7. German message 2.
8. (Optional) Trailer message.

# Assigning Batch Publishing Rules

All applications can use common, centralized tables and pages to define how to publish incremental messages from an application program. The Publish utility uses batch publish rules to process the data from the application program.

This section discusses how to:

- Associate a rule to a message and characterize the rule.
- Map a message record to another record.
- Assign an application program to a publishing rule.

**Note.** You can link application programs to multiple publishing rules for the same message or different messages. The Publish utility treats each publishing rule as a separate publishing cycle.

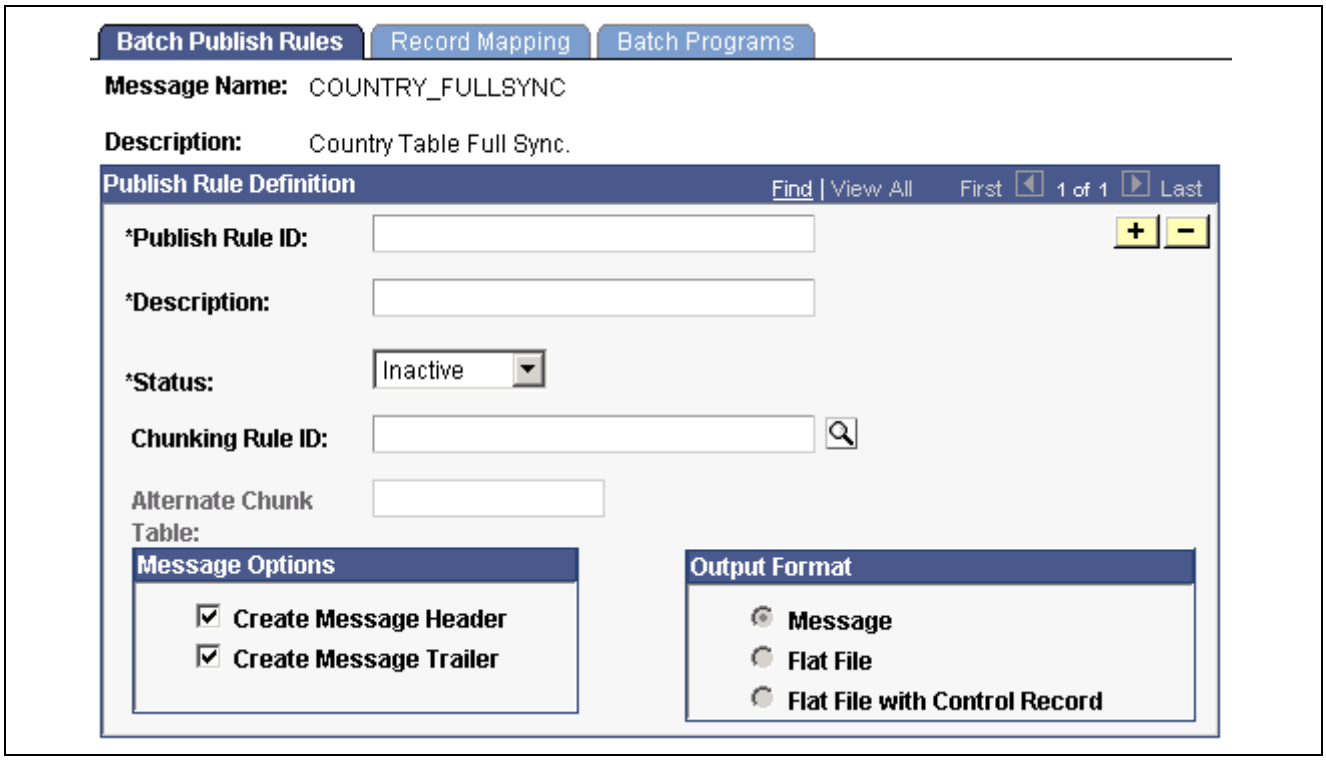
## Pages Used to Assign Batch Publishing Rules

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Batch Publish Rules | EO_MSGPUBATCH | Enterprise Components, Integration Definitions, Batch Publish Rules | Associate a rule to a message and characterize the rule. |
| Record Mapping | EO_MSGRECMAP | Enterprise Components, Integration Definitions, Batch Publish Rules, Record Mapping | Map a message record to another record. |
| Batch Programs | EO_MSGBATPGM | Enterprise Components, Integration Definitions, Batch Publish Rules, Batch Programs | Assign an application program (PROCESS_NAME) to the publish rule. |

## Associating a Rule to a Message

Access the Batch Publish Rules page.

Copyright © 1992-2006, Oracle. All rights reserved.




Batch Publish Rules page

**Publish Rule ID** Enter the rule ID to associate with the publish rule that you define.

**Status** Select *Active*if the publish definition is valid.

**Chunking Rule ID** Associate a chunking method with the publish rule, if needed.

## Mapping a Message Record to Another Record

Access the Record Mapping page.

**Message Record Name** Enter the name of the record in the message that you want to map to another record.

**Source/Order By Record Name** Enter the record name that the Publish utility uses to select data.

This page specifies the source data for a record in a message. It works in the same manner and accomplishes the same purpose as the Record Mapping page for a full table publish.

## Assigning an Application Program to a Publishing Rule

Access the Batch Programs page.

**Process Name** Enter the name of the COBOL, Structured Query Report (SQR), or Application Engine program that is marking the records to be published.

The Publish utility initially receives the process name from the batch parameter record that is created by the application program. The program then retrieves and processes each publish rule for the application process name. The process

 Copyright © 1992-2006, Oracle. All rights reserved.

name can be any 12-character string, as long as it matches what the application program inserts into the batch parameter record.

If you select a flat file format, the Publish utility does not create a header or trailer message, and the utility ignores any chunking rules. Instead, the utility creates a single flat file.

# Setting Up Message Chunking

This section provides an overview of message chunking and discusses how to:

- Identify when to use chunking.
- Select chunking fields.
- Create chunking rules.
- Define the chunking rule description.
- Maintain chunking data for business units.
- Maintain chunking data for setIDs.
- Add nodes to existing chunking rules.
- Assign business units or setIDs to a chunking rule.
- Assign chunking rules to a business unit.
- Assign chunking rules to a setID.
- Create custom chunking tables.

Copyright © 1992-2006, Oracle. All rights reserved.

## Pages Used to Set Up Message Chunking

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Chunking Rule | EO_CHUNKRULE | Enterprise Components, Map Chunking Rules, Define Chunking Rule | Define a chunking rule description. |
| BusUnit Mapping (business unit mapping) | EO_CHUNKBU | Enterprise Components, Map Chunking Rules, Business Units | Maintain chunking data for business units. |
| SetId Mapping | EO_CHUNKSETID | Enterprise Components, Map Chunking Rules, SetIds | Maintain chunking data for setIDs. |
| Add Nodes to Chunk Rule | EO_ADNODECHUNK_PNL | Enterprise Components, Map Chunking Rules, Node to ChunkRule | Add nodes to existing chunking rules. |
| Quick Map | EO_ADDSIDNODE_PNL | Click the Add button on the Add Notes to Chunk Rule page. This button is available after you have added nodes to the chunking rule. | Assign business units to a chunking rule. |
| Map Business Unit | EO_ADDNODEBU_PNL | Enterprise Components, Map Chunking Rules, ChunkRule/Node to BU | Assign chunking rules to a business unit. |
| Map Set IDs | EO_ADDNODESID_PNL | Enterprise Components, Map Chunking Rules, ChunkRule/Node to Setid | Assign chunking rules to a setID. |

## Understanding Message Chunking

If you publish to multiple nodes, you might want the messages to be routed based on a specific field. Chunking rules direct the message. You can use the same chunking fields for breaking a large message into smaller messages, as well as for associating messages with a node (based on those same fields). You set up this kind of relationship between nodes and the fields used to break the message apart (break fields) by using chunking message pages.

You can, depending on some XML content-based logic, use message chunking to route and deliver groups of transactions to different third-party nodes.

For example, consider purchase orders. If you run the full batch publish, each third-party node receives an XML message containing all purchase orders that are dispatched, regardless of whether any purchase orders are intended for that particular customer. With chunking, however, you can set up a chunking rule to chunk a message by customer ID. This creates an XML message for each customer that contains only purchase orders intended for that particular customer.

---

**Note.** Do not confuse message chunking with channel partitioning. You use channel partitioning to partition a channel by a level zero key field. If a field exists on level zero of the record in the message by which you can uniquely distinguish and group transactions to be processed in parallel, partitioning the message by this field increases performance. Without partitioning, a PeopleSoft subscribing system must process each incoming message in the order in which the message is received according to the Publication ID (PUBID) field.

---

 Copyright © 1992-2006, Oracle. All rights reserved.

# Identifying When to Use Chunking

Use chunking when:

- The message data is large, and the subscriber is consistently interested only in part of the data.
- Subscribers can more efficiently process the message data by chunking messages.
- You have trading-partner-specific content and legally do not want data to be shared among vendors.

You can chunk messages by:

- Locations and inventory shipments by business unit.
- Customers by setID.
- Employees by department or company.
- Sales order acknowledgements by setID and customer ID.
- Purchase orders and purchase order changes by setID and vendor ID.

# Selecting Chunking Fields

To maximize performance and prevent application developers from maintaining complicated views of the data, create staging or temporary tables that contain the chunking fields as the highest order key fields.

Chunking fields can affect performance and alter the options that are available to the Publish utility. The Publish utility creates SQL for each table that is defined in the message object. Tables that are defined in the message can be mapped to an alternative source table or viewed on the Record Mapping page under Publish Rule Definition page.

The source table (or view) serves two purposes:

- It enables the data that must be published to come from a source other than the table that is defined in the message.
- It enables the data to be ordered so that the Publish utility can process rows in the correct sequence.

Because the SQL is run only once and includes a subquery against the values in the chunking table, you must define all chunking fields in every table that is used to retrieve data for the message. The SQL order-by clause is set according to the key fields that are defined in the table. The result is that chunking fields must be defined as key fields for the Publish utility to work.

## Example of Generated SQL for Chunking

This is an example of SQL that is generated for chunking:

```
Select * from PS_INV_ITEMS A
where EXISTS (Select 'Y' from PS_EO_SETID_EOC B
where B.CHUNK_RULE_ID = 'SETID'
and B.EFFDT = '20000201'
and A.SETID = B.SETID)
 order by A.SETID, INV_ITEM_ID, EFFDT
```

The field that you select to chunk on determines the view table that you must create:

Copyright © 1992-2006, Oracle. All rights reserved. 

| Chunking Field Attribute | Corresponding View Table |
|---|---|
| Chunking field is a key field in level zero table. | By rule, the chunking fields are also key fields in the child tables. The key fields of a parent table must be key fields in the child table and in the same order. If the chunking fields are not the highest order key fields, create a view that consists of all fields in the source table, with the chunking fields as the highest order key fields, followed by the remaining key fields from the source table. Then map this view table to the source table on the Record Mapping page under Publish Rule Definition page. |
| Chunking field is not a key field. | Create a view of the source data that consists of all fields in the source table, with the chunking fields as the highest order key fields, followed by the rest of the key fields from the source table. Then map this view table to the source table on the Record Mapping page under Publish Rule Definition page. |
| Chunking field is not in a source table. | Create a view table that joins the source table to an existing table that contains the chunking fields. This view must consist of all fields in the source table and the chunking fields from the joined table. The chunking fields are the highest order key fields, followed by the rest of the key fields from the source table. Then map the view table to the source table on the Record Mapping page under Publish Rule Definition page. |

**Note.** When running a batch publish rule, the Publish utility runs cleanup logic, which either updates fields or deletes rows in the source tables. If the source table is a view that contains a join, then the option to delete published rows fails.

## Example

The following sample SQL code creates a view table that joins the PS_OMEC_CP_OPT_DET source table to an existing PS_OMEC_HDR_OUT table that contains the chunking fields. The B.SETID_CUSTOMER and the B.CUST_ID chunking fields are the highest order key fields from the joined table (PS_OMEC_HDR_OUT), followed by the rest of the key fields from the source table (PS_OMEC_CP_OPT_DET).

```
SELECT B.SETID_CUSTOMER
, B.CUST_ID
, A.BUSINESS_UNIT
, A.ORDER_NO
, A.ORDER_INT_LINE_NO
, A.CP_MODE
, A.CP_COMP_SEQ
, A.OPTION_NAME
, A.OPTION_VALUE
, A.OPTION_DESC
, A.VAR_TYPE
, A.VAR_LENGTH
```

 Copyright © 1992-2006, Oracle. All rights reserved.

```
, A.VAR_DECIMAL
, A.PROCESS_INSTANCE
, A.AUDIT_ACTN
, A.IN_PROCESS_FLG
FROM PS_OMEC_CP_OPT_DET A
, PS_OMEC_HDR_OUT B
WHERE A.BUSINESS_UNIT = B.BUSINESS_UNIT
AND A.ORDER_NO = B.ORDER_NO
```

## Creating Chunking Rules

The chunking rule consists of four tables:

| Level | Table | Description |
|---|---|---|
| Level 0 | EO_CHUNKRULE | A system table delivered with live data. |
| Level 1 | EO_CHUNKEFFDT | A system table delivered with live data. When a chunking rule is saved, a row is added to this table with the effective date (EFFDT) field automatically populated from the current date and the effective status set to*Active*. |
| Level 2 | EO_CHUNKNODE | This is not a system table and is delivered empty. |
| Level 3 | *NAME_EOC* | A user-defined chunking table. |

All user-defined chunking table names must end in *_EOC*. PeopleSoft provides two standard tables: EO_BUSUNIT_EOC for business unit values and EO_SETID_EOC for setID values. The different types of user-defined chunking tables are:

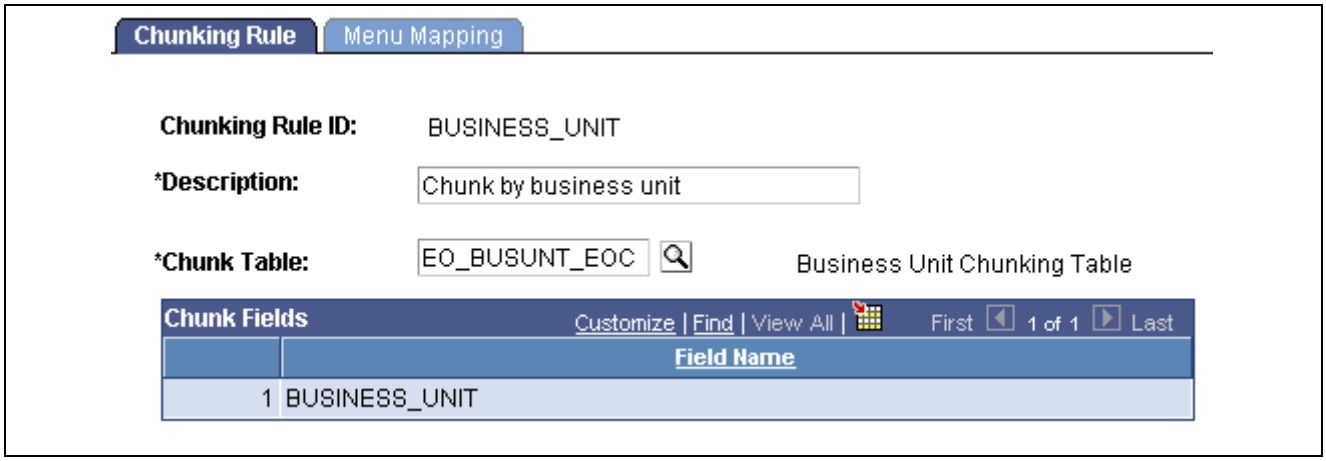
| Table Type | Description |
|---|---|
| Derived Table | Contains only the chunking fields. Can be used by the Publish utility to chunk the message whenever the value of the chunking field changes. In derived tables, there is no relationship between the value of the chunking fields and message node names that are used to route the message. OnRoute PeopleCode needs hard-coded routing logic or additional tables to route the message to the appropriate nodes. |

Copyright © 1992-2006, Oracle. All rights reserved.

| Table Type | Description |
|---|---|
| SQL Tables | Contains the fields:<br>• CHUNK_RULE_ID<br>• EFFDT<br>• MSGNODENAME<br>• Chunking fields<br>Limits the published data to the values of the chunking fields in the chunking table and contains the message node name that is used to route the message. |
| Alternate Chunking Tables | Enables reuse of existing chunking tables. Must end in *_EOV*. |

## Defining the Chunking Rule Description

Access the Chunking Rule page.




Chunking Rule page

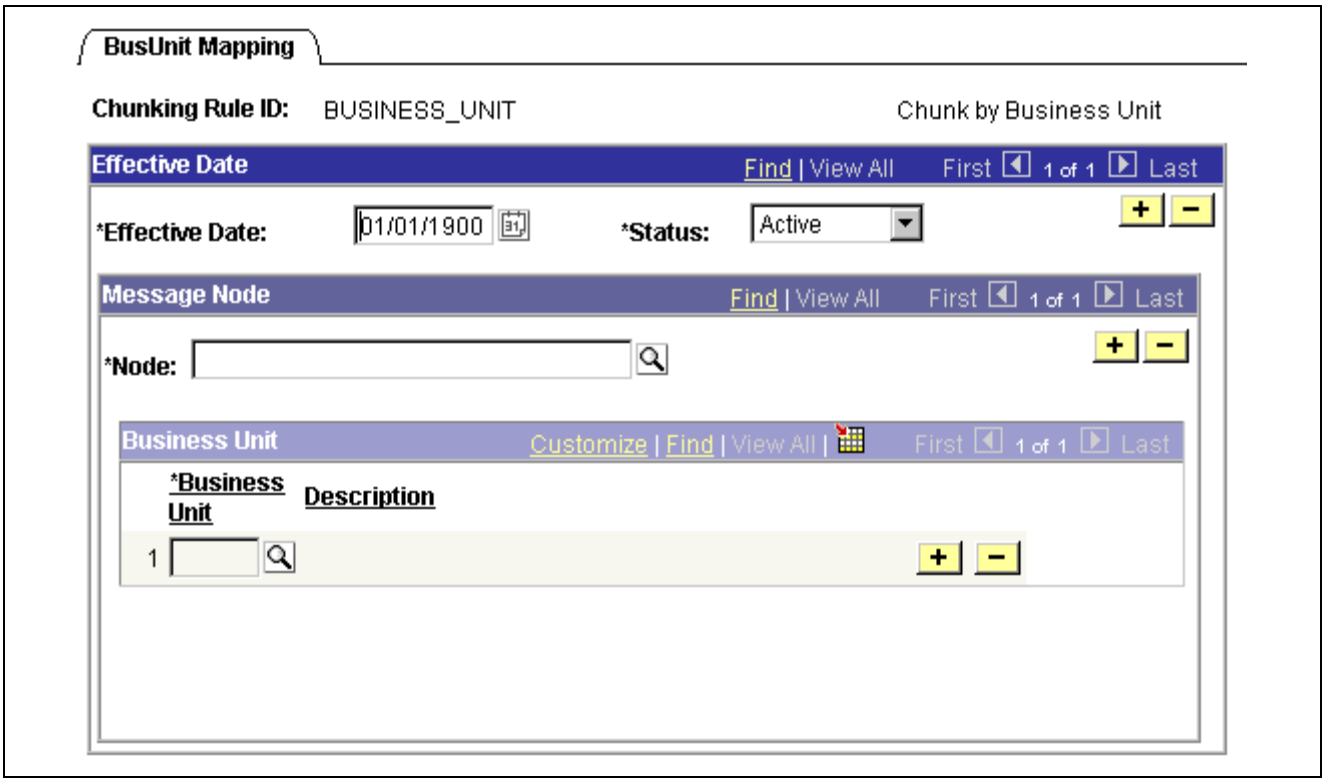
Chunk field are the chunking fields that are defined in the Chunk Tablefield.

The chunking fields appear in this scroll area to verify that the correct chunking table was entered. PeopleSoft provides chunking rules for business unit and setID for all application databases. Adding a new chunking rule inserts a row into the EO_CHUNKEFFDT table, with a default effective date (EFFDT) of the current date.

## Maintaining Chunking Data for Business Units

Access the BusUnit Mapping page.


Copyright © 1992-2006, Oracle. All rights reserved.




BusUnit Mapping page

All four levels of the chunking rule tables appear.

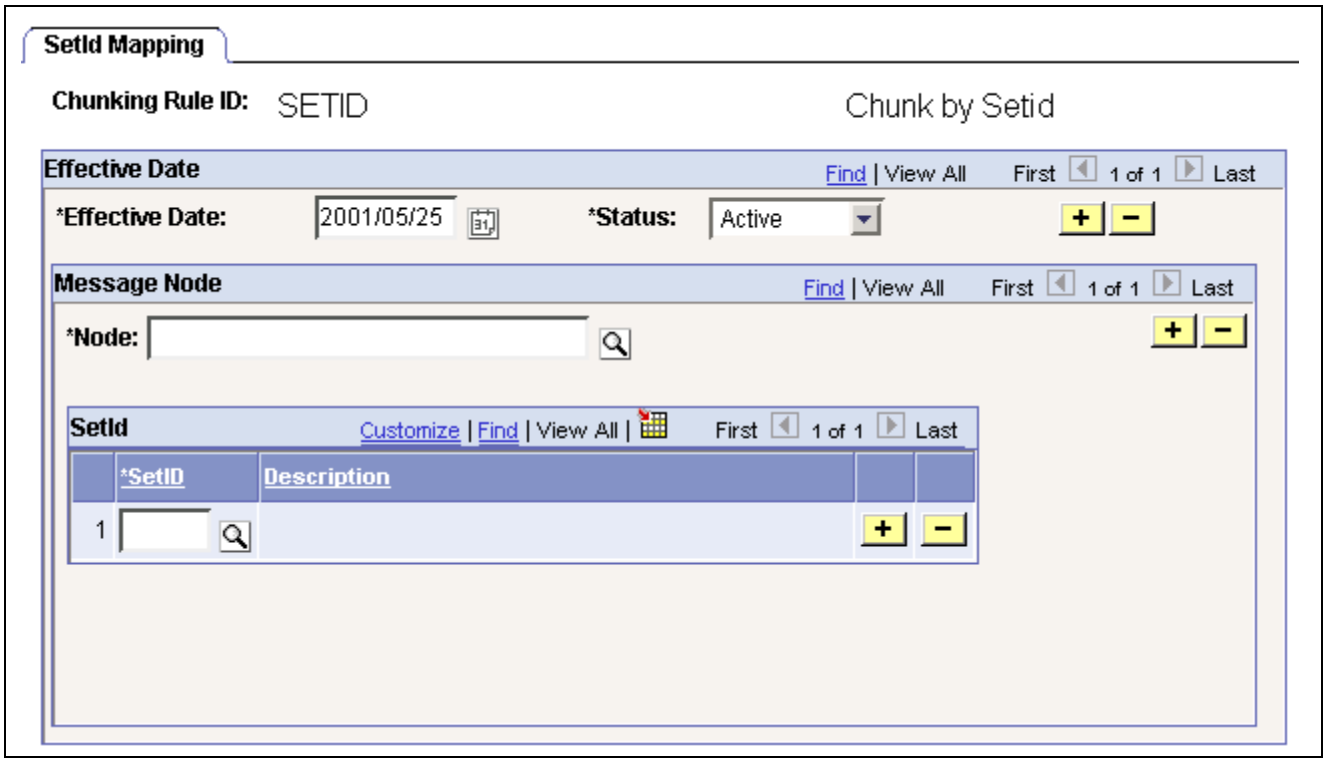
PeopleSoft provides chunking tables for business unit and setID that are maintained by a series of components (such as components that are created for maintaining the business unit chunking table).

You can use each component to update the underlying relationship between the business unit and the subscribing nodes. You can maintain the data either by business unit or by node, individually or as a group, to reduce the amount of entry work.

## Maintaining Chunking Data for SetIDs

Access the SetId Mapping page.

Copyright © 1992-2006, Oracle. All rights reserved.




SetId Mapping page

All four levels of the chunking rule tables appear.

PeopleSoft provides chunking tables for business unit and setID that are maintained by a series of components (such as components that are created for maintaining the business unit chunking table).

You can use each component to update the underlying relationship between the business unit and the subscribing nodes. You can maintain the data either by business unit or by node, individually or in a group, to reduce the amount of entry work.

## Adding Nodes to Existing Chunking Rules

Access the Add Nodes to Chunk Rule page.

To add nodes to an existing chunking rule:

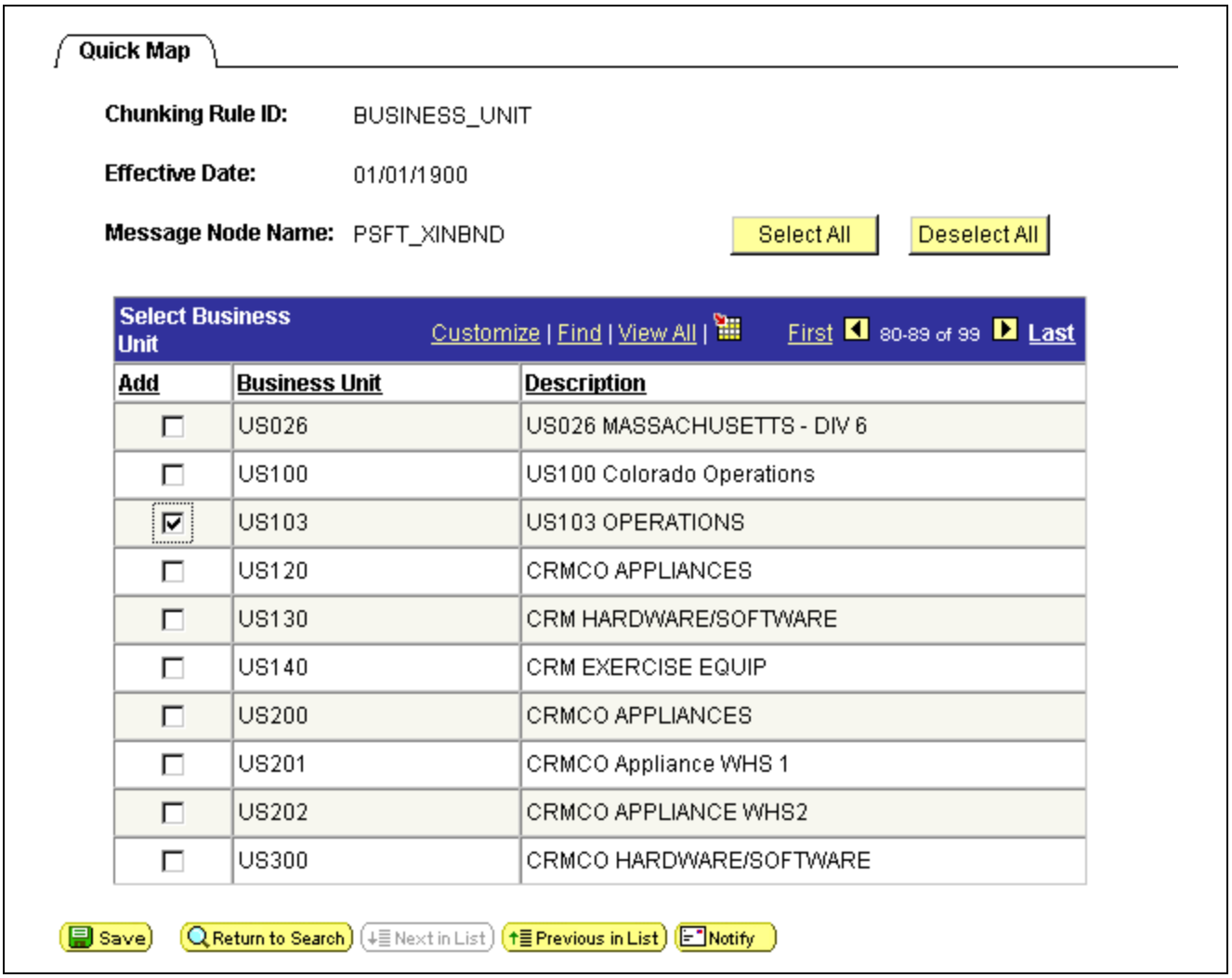
1. Select the check box in the Add column of the nodes that you defined earlier.
2. Click the Save button to display the Add Chunk Values column.
3. Click the Add button in the Add Chunk Values column for the nodes that you want to add.

   The Quick Map page appears.

## Assigning Business Units or SetIDs to a Chunking Rule

Access the Quick Map page.


Copyright © 1992-2006, Oracle. All rights reserved.




Quick Map page

If you previously accessed business unit chunking rules, you can add business units to a chunking rule. If you previously accessed setIDs, you can add setIDs to a chunking rule.

---

**Note.** You cannot access the Quick Map page without first using either the BusUnit Mapping page or the SetId Mapping page to add an effective-dated node to the chunking rule ID.

---

**Select All** and **Deselect All** Click to add or remove all business units that are assigned to the node. Add check boxes are selected for business units that are assigned to the node.

## Assigning Chunking Rules to a Business Unit

Access the Map Business Unit page.

Copyright © 1992-2006, Oracle. All rights reserved.

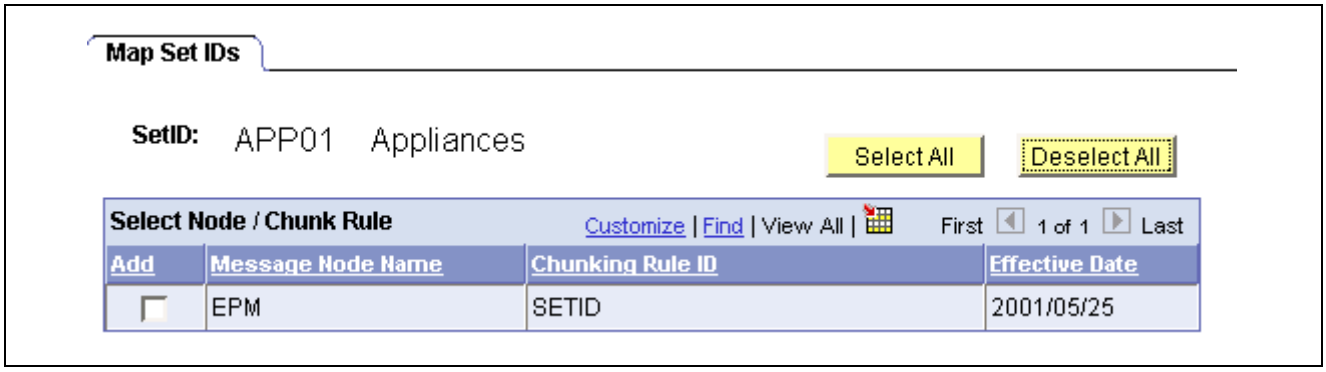



Map Business Unit page

**Select All** and **Deselect All** Click to add or remove all nodes that are assigned to the business unit. Add check boxes are selected for nodes that are assigned to the business units.

## Assigning Chunking Rules to a SetID

Access the Map Set IDs page.




Map Set IDs page

**Select All** and **Deselect All** Click to add or remove all message nodes that are assigned to the setID. Add check boxes are selected for message nodes that are assigned to the setIDs.

# Creating Custom Chunking Tables

This section discusses how to:

- Create a custom chunking table.
- Create a view for the component search record.
- Create maintenance pages.
- Create a component.
- Create routing PeopleCode.


Copyright © 1992-2006, Oracle. All rights reserved.

## Creating a Custom Chunking Table

To create a custom chunking table:

1. Select File, Open in PeopleSoft Application Designer.
2. Select *Record* in the Definition drop-down menu.
3. Open the EO_BUSUNT_EOC record.
4. Save the record as *YOUR_TABLE*_EOC.
5. Remove the BUSINESS_UNIT field.
6. Insert the custom chunking fields at the bottom of the record.
7. Select File, Save.
8. Build the SQL table.

## Creating a View for the Component Search Record

To create a view for the component search record:

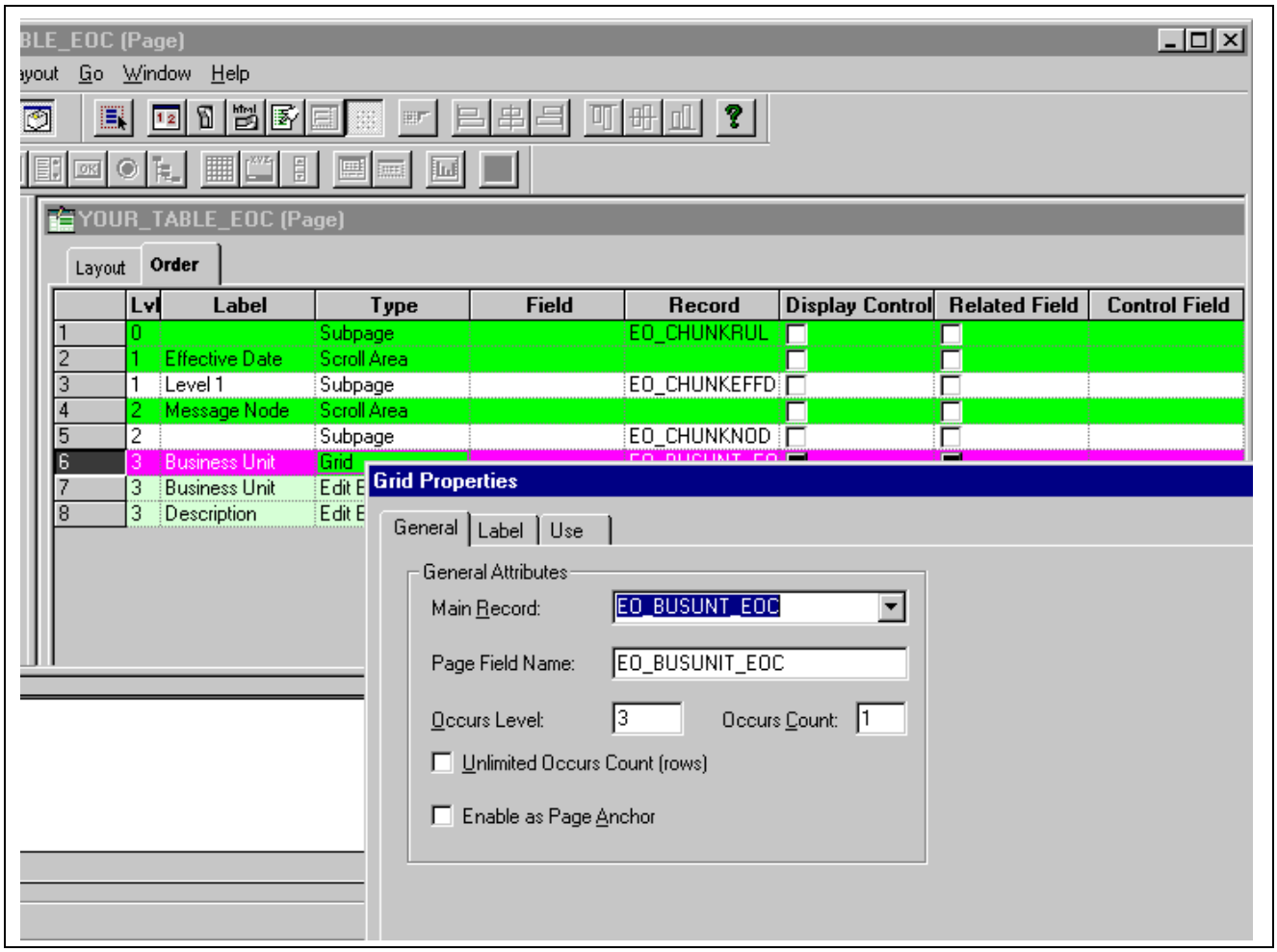
1. Select File, Open in PeopleSoft Application Designer.
2. Select *Record* in the Definition drop-down menu.
3. Open the EO_CHUNKBU_VW record.
4. Save the record as *YOUR_TABLE*_VW.
5. Select the Record Type tab.
6. Open the SQL editor.
7. Modify the Where clause.

   Change `WHERE RECNAME_CHUNK=EO_BUSUNT_EOC` to `WHERE RECNAME_CHUNK=YOUR_TABLE_EOC`.
8. Select File, Save.
9. Build the SQL view.

## Creating Maintenance Pages

You can also create maintenance pages.

Copyright © 1992-2006, Oracle. All rights reserved. 




Grid Properties dialog box

To create maintenance pages:

1. Select File, Open in PeopleSoft Application Designer.
2. Select *Page* in the Definition drop-down menu.
3. Open the EO_CHUNKBU page.
4. Save the page as *YOUR_PAGE*.
5. Select the Order tab.
6. In the Type column, double-click Grid to open the Grid Properties dialog box.
7. Change the value in the Main Record and Page Field Name fields to *YOUR_TABLE_EOC*.
8. Click OK.
9. Delete the business unit and description columns.
10. Add chunking fields from YOUR_TABLE_EOC.
11. Select File, Save.

## Creating a Component

To create a component:


Copyright © 1992-2006, Oracle. All rights reserved.

1. Select File, New in PeopleSoft Application Designer.
2. Select *Component* in the New Definition box.
3. Select Insert, Page Into Component.
4. Enter *YOUR_PAGE.*
5. Click Close.
6. Select File, Definition Properties
7. Select the Use tab to edit component properties:
    a. In the Search record field, enter *YOUR_TABLE_VW.*
    b. Select the Update/Display, Update/Display All,and Correction check boxes.
8. Click OK.
9. Select File, Save As, and save the page group as *YOUR_COMPONENT.*
10. Add YOUR_PAGE_GROUP to YOUR_MENU that is used by your application.

## Creating Routing PeopleCode

OnRouteSend (and OnRouteReceive) are PeopleCode methods that are tied to the message for routing, based on the message contents. If you want the contents of the message (such as a message chunking field value) to determine the subscribing nodes that should receive the message, OnRouteReceive PeopleCode must contain the logic to examine the message and return a list of subscribing nodes.

The OnRouteSend and OnRouteReceive methods are contained in the IRouter application class. The IRouter application class is located in PeopleSoft Application Designer in the Integration subpackage of the PS_PT application package.

PeopleCode functions provided by common components, GetNodes and RtnNodes, work with any message and chunking rule. For a given message, these nodes select the chunking rule for the publish rule that is assigned to the message.

The functions do the following:

- Build SQL based on the chunking fields as defined in the chunking table.
- Extract chunking field values from the message.
- Run the associated SQL.
- Compare the array of nodes returned to the application server against the list of nodes for the message channel.
- Create a publish contract for nodes in both arrays.

  You can override the publish rule from the message, specifying an optional parameter when calling the functions.
- Return an array of nodes that is based on the nodes that are assigned to the message channel if the publish rule is invalid or does not contain a chunking rule.

  Returning an array of nodes enables the functions to work regardless of whether chunking is set up for the publish rule.

To route any message that uses chunking, use generic PeopleCode functions.

These functions are called from routing PeopleCode:

Copyright © 1992-2006, Oracle. All rights reserved. 

| | |
|---|---|
| **GetNodes** | Returns an array of nodes to the application server. |
| **RtnNodes** | Returns an array of nodes to the calling PeopleCode. |

These functions are internal functions:

| | |
|---|---|
| **FndNodes** | Builds an array of nodes for the message. |
| **GetPubRule** | Selects the chunking rule for the publish rule. |
| **GetChunkInfo** | Selects the chunk table for the chunking rule. |
| **BuildSQL** | Builds SQL to select nodes from the chunking table for specific chunking field values from the message. |
| **GetValue** | Gets the chunking field values from the message. |
| **HasNodes** | Determines whether a chunking field is mapped to any nodes for a particular chunking rule. |

The following code example shows the logic that you can add to SavePostChange PeopleCode for the Customer_General component to verify if the setID can publish the message by calling the HasNodes() function:

```
Declare Function HasNodes PeopleCode FUNCLIB_EOEIP.PUBLISH_ROUTE_PC
FieldFormula;
Local Message &MSG;
Local Rowset &RS0;
Local string &PublishRule;
&MSG = CreateMessage(MESSAGE.CUSTOMER_MSG);
/* Check if message is active */
If &MSG.IsActive Then
     &RS0 = GetLevel0();
     &PublishRule = "CUSTOMER_SYNC";
     /* Call function passing publish rule and rowset, which returns
     true if this setID can publish the message */
     If (HasNodes(&PublishRule, &RSO)) Then
          &RS0 = GetLevel0();
          &MSG.CopyRowsetDelta(&RS0);
          &MSG.Publish();
     End-If;
End-If;
```

 Copyright © 1992-2006, Oracle. All rights reserved.

# CHAPTER 19

# Using the Error Handling Utility

This chapter provides an overview of the Error Handling utility and discusses how to:

- Set up and maintain message errors.
- Correct message errors.

**Note.** The Error Handling Utility is now superseded by the Services Operations Monitor that is part of PeopleSoft Integration Broker.

## Understanding the Error Handling Utility

The Error Handling utility is a PeopleTools application that you use to view and correct messages that are received by the subscriber. You can also use this utility to correct data that is stored in staging tables.

**Note.** The Service Operations Monitor, available in the PeopleTools 8.48 release in PeopleSoft Integration Broker, supercedes the Error Handling utility.

**Note.** If you use the Error Handling utility, you should be familiar with messaging technology.

### Error Management Process

PeopleSoft applications that receive flat file data from other systems through batch processes have built-in facilities to validate and correct data prior to updating the main application tables. Likewise, before updating core PeopleSoft application tables, the subscription process detects data errors in the messages that it receives. These error messages are stored in either message queues or staging tables.

In some cases, however, errors can't be sent back to the third party for correction (such as when data is in a flat file). In these cases, you must provide error processing on incoming data so that messages that contain information on business objects, such as items and vendors, can be corrected and reprocessed in the PeopleSoft system.

In many integration point implementations, especially those that involve huge transactions and complex data validations, subscription codes are written to run simple incoming data validations. Upon a successful outcome, the system performs these steps:

1. Writes incoming data to staging tables.
2. Runs more stringent data validation processes, usually in batch processes.

As a result of the first subscription process, entries are written to a special staging table (EO_EIP_CTL). The EO_EIP_CTL table is keyed by a single key, EIP_CTL_ID, and has messaging keys. The EO_EIP_CTL table maintains links between the source message and the staged data and ensures that the data maintenance program can identify the staging tables.

Copyright © 1992-2006, Oracle. All rights reserved.

To find the particular message that you want to view:

1. Select a message by using standard selection criteria, including data maintenance programs, business unit or setID, data status, and staging table or message queue.
2. Click a detail button to access the application page related to the edited message.
3. Make your edits.
4. Save the message.

---

**Note.** Messages that are selected from a staging table are saved to the staging table. Before they're saved permanently (to the application table), the corrected messages that were selected from the message queue are assigned a status of *Changed* and saved to the same queue. The messages then undergo the subscription process data validation again, after which they're saved to an application table.

---

A data maintenance program corrects errors in data that enters the PeopleSoft system.

---

**Note.** Subscription error management does not apply to real-time data that enters the PeopleSoft system through PeopleSoft Business Interlinks.

---

Error management consists of the following sequential activities:

1. The PeopleSoft system receives XML messages from a third-party source.
2. Subscription processes validate the data and send the data to:
   - PeopleSoft application tables (if no errors were found).
   - Message queue (if errors were found).
   - PeopleSoft staging tables (if subscription logic requires).
3. The Error Handling utility interacts with:
   - Message queue.
   - Staging tables.
4. The user interacts with:
   - The Error Handling page (to select a message for review or correction).
   - Application pages (for editing data errors).
5. The PeopleSoft system updates:
   - Message queue (if the original message was selected from the queue).
   - Staging tables (if the original data was extracted for staging tables).

# Setting Up and Maintaining Message Errors

This section discusses how to:

- Create error-correction pages and the necessary hooks to the error handling utility.
- Set up the error handling utility.
- Set up row security.

 Copyright © 1992-2006, Oracle. All rights reserved.

- Set up workflow notification in PeopleSoft Application Designer.
- Test the error handling utility.

## Pages Used to Set Up and Maintain Message Errors

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Data Maintenance | EO_EIP_CTL_SETUP1 | Enterprise Components, Integration Definitions, Data Maintenance Utility | Set up the Error Handling utility and correct message errors. |
| Workflow/Security | EO_EIP_CTL_SETUP2 | Enterprise Components, Integration Definitions, Data Maintenance Utility, Workflow/Security | Set up row security. |
| EO EIP Data Maint (EO EIP data maintenance) | EO_EIP_CTL_MAINT | Enterprise Components, Integration Definitions, Review Centralized Error | Test the error handling utility.<br>Test subscription processes must error out. Create a message subcontract with *Error* status, or create a stage-based incoming transaction with a status value of 1 *(Error)*.<br>**Note.** Error conditions depend on processes that are specific to the application. |

## Creating Error-Correction Pages

Use PeopleSoft Application Designer to create error-correction pages and components.

---

**Note.** The Header Details page and the Error Details page are applicable only if data resides in the messaging queues. Use one Header Details page and one Error Details page for each field.

---

When creating error-correction pages, follow these guidelines:

- Use EO_EIP_CTL as the main header table if you use staging tables.

  Developers can use unique EIP_CTL_ID generator functions (Generate_EIP_CTL_ID and Increment_EIP_CTL_ID) available in the FUNCLIB_EOEIP.EIP_CTL_ID FieldFormula event.

- Use the generate EIP_CTL_ID function to generate a unique key value for EIP_CTL_ID, based on the subscription process instance.

  The EIP_CTL_ID function is the sole key in the EO_EIP_CTL header table.

- Hook to EO_EIP_CTL to use the Error Handling utility for staging-table-based error handling.

  See Chapter 19, "Using the Error Handling Utility," Setting Up the Error Handling Utility, page 218.

- Modify the record PeopleCode in the derived work record EO_EIP_CTL_WRK.

  The derived work record called EO_EIP_CTL_WRK contains most of the processing logic for the utility. Although most of the codes are generic, you must write application-specific codes for the return button field (EIP_RETURN_BTN).

Copyright © 1992-2006, Oracle. All rights reserved. 

- Add codes to return the user to the main Error Handling page.

  Use the EO_EIP_CTL_WRK derived record in your component and write component PeopleCode for the record to handle any unique requirements.

  **Note.** This unique code belongs only to your component and cannot be shared.

### FUNCLIB_XXEIP Codes

Functions are created and stored in product-specific FUNCLIB_XXEIP records, with *XX* representing the product. Application developers can look up these function libraries for possible use as templates.

Common integration-related functions are placed in the FUNCLIB_EOEIP record.

### EIP_DETAIL_BTN FieldFormula

Function Copy_Detail_Errors (&WRK_FIELD, &J, &WRK_ROWSET) copies edit errors to a generic error subpage.

Function BuildQueueRowset (&WRK_ROWSET1 As Rowset, &WRK_ROWSET2 As Rowset, &MSG_ROWSET As Rowset, &SCROLL, &RECORD_FROM, &RECORD_TO) generically builds the queue-based transfer page. This function is useful for single record messages.

### FUNCLIB_EOEIP.EIP_CTL_ID FieldFormula

You need an EIP_CTL_ID every time that you process a message to maintain error handling. Use Function Generate_EIP_CTL_ID and Increment_EIP_CTL_ID to create EIP_CTL_IDs based on a new subscription process instance. For the Generate_EIP_CTL_ID function, invoke method 4 from the list of process instances.

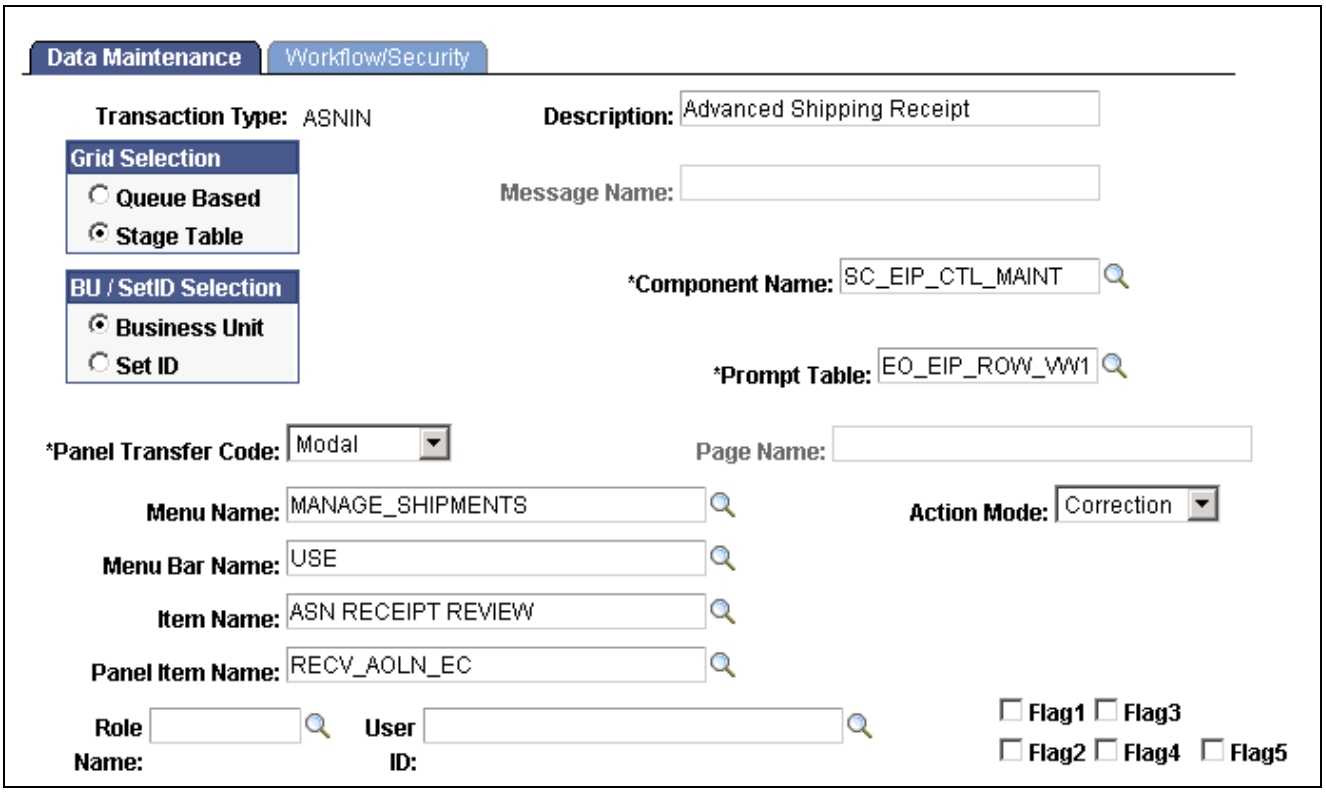
**Note.** Although in some cases a random number generator is used for creating a new EIP_CTL_ID by calling Generate_EIP_CTL_ID(1), you should use the subscription process instance if possible. Generate_EIP_CTL_ID(4) uses the message subscription process identifier number. Remember to activate this on the message subscription properties by selecting the Generate Subscription Process Instance check box.

***Warning!*** The Generate_EIP_CTL_ID and Increment_EIP_CTL_ID functions call other functions in FUNCLIB_ININTFC (an FDM record). You must recreate these additional functions whenever you use Generate_EIP_CTL_ID and Increment_EIP_CTL_ID within applications.

## Setting Up the Error Handling Utility

Access the Data Maintenance page.

 Copyright © 1992-2006, Oracle. All rights reserved.




Data Maintenance page

**Queue Based** and **Stage Table** — PeopleTools messages are queue-based; bulk data or application data is usually staged.

**Business Unit** and **SetID** — Select a business unit or setID on which to base the data.

**Component Name** — Select the name of the component for accessing the Error Handling/Data Maintenance page (EO_EIP_CTL_MAINT). This is the component that the user opens for error-correction activities. You should select *EIP_DTA_CTL*. This component uses the same fonts and images for all incoming data maintenance or error-correction activities within PeopleSoft for non-real-time integration.

**Prompt Table** — Follow the navigation path to use the Error Handling utility, and select a prompt table for the data maintenance program (EIP_PROGRAM). The PeopleSoft system includes a common error handling menu structure that points to the main component EIP_DTA_CTL. Based on user privileges, the Error Handling utility prompts from a selected list of data maintenance programs.

**Panel Transfer Code** — Select *Next Panel* to receive a transfer panel name prompt, which gives you a selection of pages that are part of the main data maintenance component that you selected in the Main Data Maint Panel Group field. Select one page.

Selecting either *Transfer* or *Modal* opens additional fields.

**Role Name** — Select a name to set up workflow notification and to notify all role users listed.

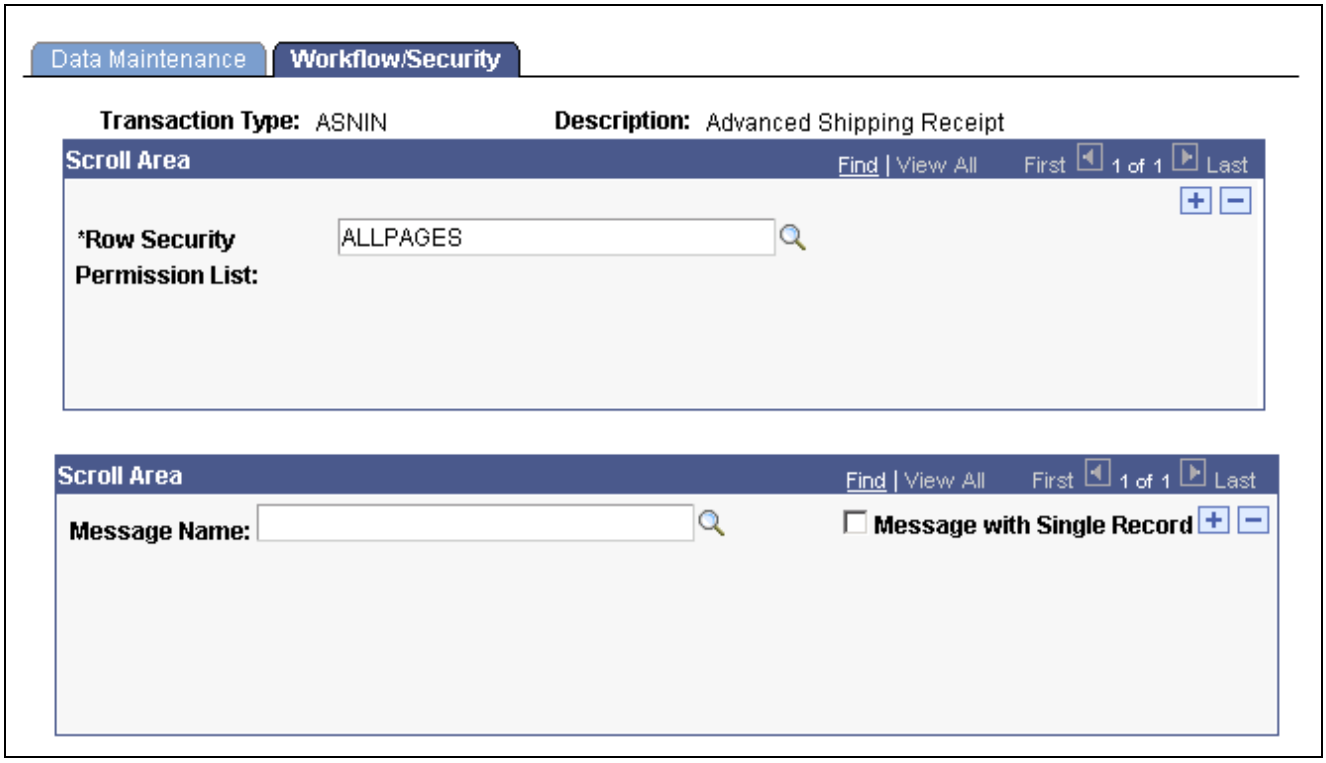
**User ID** — Select to notify only specific role users.

Copyright © 1992-2006, Oracle. All rights reserved.

**Flag1, Flag2, Flag3, Flag4,** and **Flag5** Users often won't see these check boxes. Application developers use them for extra coding flexibility. Developers must create documentation for these check boxes to use them.

## Setting Up Row Security

Access the Workflow/Security page.




Workflow/Security page

**Row Security Permission List** Assign a row security class to the transaction type that you're setting up.

**Note.** The value that you select determines which user can see the given transaction type. You can enter multiple row security classes.

## Setting Up Workflow Notification in PeopleSoft Application Designer

You can set up an EO_WF_ERR Application Engine process to scan the EO_EIP_CTL table periodically to look for rows with *ERROR* status. When the system finds errors, it generates workflow notifications and routes them to the role users that you designated in the User ID field on the Data Maint Setup1 page. Clicking the worklist item in the workflow notification transfers the user to the Error Handling/Data Maint (error handling/data maintenance) page.

To set up workflow notification:

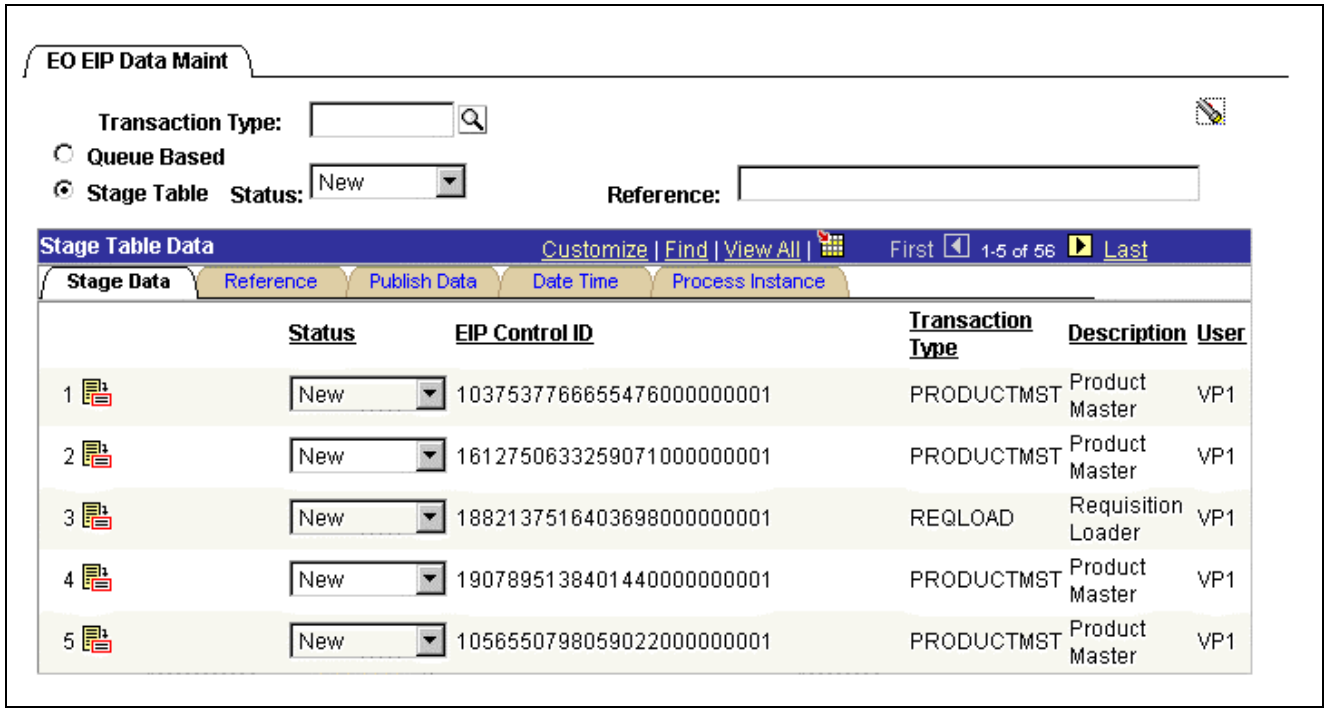
1. Open an instance of PeopleSoft Application Designer.
2. Select File, Open.
3. Select *Business Process* from the Definition drop-down menu.


Copyright © 1992-2006, Oracle. All rights reserved.

4. Open business process EC_MANAGE_ERRORS.
5. Right-click the MANAGE ERRORS icon and select *View Definition.*
6. Double-click the Correct Errors icon to open the Step Definition dialog box.
7. Click the Attributes button to open the Step Attributes dialog box.
8. Complete the required navigation information to the error correction page in the Step Attributes dialog box and click OK.

## Testing the Error Handling Utility

Access the EO EIP Data Maint page.




EO EIP Data Maint page

**Queue Based** and **Stage Table** — Select whether the errors you want to check are queue-based or stored in a staging table.

**Status** — Select from *Cancelled, Complete, Error, Hold, In Process, Incomplete, New,* and *Reprocess.*

 — Click the Show Detail Entry button to transfer to the application page for necessary error-correction activities.

# Correcting Message Errors

This section discusses how to correct message errors.

Copyright © 1992-2006, Oracle. All rights reserved.

## Understanding the Workflow Notification Process

Workflow error notification involves:

1. A process-scheduled Application Engine program scans data errors in the staging tables (or in the Subscription Contract Message queues) at a given interval.

   **Note.** You can also run a separate Workflow Notification Application Engine program for messaging queues.

2. When the Application Engine program finds errors in the actual message queue or the staging summary table, it opens respective component interfaces that invoke the designated temporary components for inserting error data into the underlying temp tables.
3. Saving the temporary page (through component interface calls) triggers workflow PeopleCode to send worklist notifications to designated users, depending on the role name or user ID that you selected on the Data Maint Setup1 page.

   **Note.** The Application Engine program traps only the first error for each field, but it can trap multiple errors for each record.

## Correcting Message Errors

To correct message errors:

1. Access the Data Maint Setup1 page to open the worklist.
2. Open the worklist.

   You are transferred to the Error Correction/Data Maint page (EO_EIP_CTL_MAINT) to correct the errors.
3. Find the error rows by using a combination of search criteria, such as transaction type, business unit or setID, and error status.
4. Click the Search icon.
5. Click the Show Detail Entry icon (to the left of each row) to select a specific transaction that represents a single message or a single row of header-level record from the staging table.
6. Correct the data.
7. Click the Return button to return to the Error Handling Summary page (EO_EIP_CTL_MAINT) and continue the session with other transactions.

**Note.** In this scenario, the corrected data does not update core PeopleSoft tables. Instead, it updates the message queue or staging tables.


Copyright © 1992-2006, Oracle. All rights reserved.

# CHAPTER 20

# Using the Effective Date Publish Utility

This chapter discusses how to:

- Perform a full data publish of current effective data.
- Publish incremental messages of current effective data.
- Publish effective-dated rows from the delay table.

## Understanding the Effective Date Publish Utility

The Effective Date Publish utility enables you to design processes to update external systems that process only current data and don't use or recognize effective dating.

When working with effective dating and effective date publishing, you need to understand the following terms:

**Current Row** — The current row is the first row of data with an effective date equal to or less than the system date. Only one row can be the current row.

**Future Rows** — Future rows have effective dates greater than the system date (usually the current date).

**Historical Rows** — Historical rows have effective dates less than the current row.

**Effective Date** — An effective date is when a table row becomes effective, or the date that an action begins. The PeopleSoft system supports the concept of effective-dated rows.

**Note.** The EFFDT field is almost always a key. Specify the descending key attribute to display the row with the most recent effective date first.

**Effective Dating** — Automated effective dating saves changed data in a staging table for subsequent processing when the effective date becomes current. (Although data can be historical, current, or future, some third-party applications may support only current data. Thus, if a future-dated row is created within the PeopleSoft system, it must be delayed before transmission to the other system.)

**Effective Sequence** — An effective sequence serves two different purposes:

- If EFFSEQ is a required field, it enables the entry of more than one row with the same effective date when paired with EFFDT. The system assigns a unique sequence number to each row that has the same effective date. It also enables the first EFFSEQ to be zero.

Copyright © 1992-2006, Oracle. All rights reserved.

- If EFFSEQ is not a required field, it is not paired with EFFDT, has no special function, and can be used as a simple sequencing field.

**Effective Status** Effective status enables the system to select the appropriate effective-dated rows, when combined with the effective date field.

**Full Data Publish** The full data publish process seeds, or initially populates or repopulates, a copy of an entire table onto a remote database or legacy system. The entire contents of the table are published to all systems that require a copy of the table. Generally, full data replication occurs with setup tables (relatively static, low-volume tables that are keyed by setID) and occurs in an asynchronous manner.

When a full copy of the table exists on the external system, an incremental update provides a mechanism to keep the copy up-to-date with changes made on the master.

**Incremental Publish** The incremental publish process sends a message that contains only the rows where the data has been modified, plus the corresponding anchoring parent and grandparent rows. When a particular transactional event occurs, an incremental update of the transaction data is sent to other systems to notify them of the changes.

**Nodes** Each node represents a publishing or subscribing system of a service operation. For example, the PeopleSoft Human Resources and PeopleSoft Financials databases are each defined as a node, even if they are both on the same server.

**Service Operation Queues** Service operation queues group messages and the nodes to which they are published, so that messages are published sequentially. Each message must belong to only one queue. Queues control the ordering of messages and define timeout parameters and error thresholds. Assign nodes to a queue when you define the message in the service operation.

**Message Chunking** Chunking automatically breaks a message into several smaller messages based on the values in one or more of the fields in the level zero record. When publishing the entire contents of a table, you can use message chunking to publish only certain sets of data, or if a particular subscriber is interested in only a portion of the table.

**Request ID** Use the request ID to specify multiple requirements within the same run control.

**Run Control** You use run controls to produce full messages for objects at the same time. Run controls also associate publish rule definitions with the scheduled full publish process run. For example, you can set up a run control to publish both customer full messages and sales order full messages on a daily schedule.

# Performing a Full Data Publish of Current Effective Data

This section discusses how to:

- Create effective-dated messages.
- Create a service operation.

 Copyright © 1992-2006, Oracle. All rights reserved.

- Define chunking rules and ordering views.
- Create publish rule definitions.
- Create run controls for the Full Data Publish program.
- (Optional) Define message routing.

For full data messages that are intended for vendors who do not handle effective dating, use the Effective Date Publish utility and a current full message to publish only those rows that are currently active. Any future-dated rows are written to the delay table.

This section discusses the process involved in a full data publish of current effective data. It uses the CUSTOMER_FULLSYNC_EFF service operation as an example, but the methods and procedures that are described here apply to creating any effective-dated service operation that contains effective-dated messages.

## Pages Used for Full Data Publish of Current Effective Data

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Chunking Rule | EO_CHUNKRULE | Enterprise Components, Integration Definitions, Map Chunking Rules, Define Chunking Rules | Define the chunking rule description. |
| Full Data Publish | EO_FULLDATAPUB | Enterprise Components, Integration Definitions, Initiate Processes, Full Data Publish | Create the run control for the Full Data Publish utility.<br>The run control associates publish rule definitions with the scheduled Full Publish process run. For example, you can set up a run control to publish both customers and sales orders full messages at the end of each day. |

## Creating Effective-Dated Messages

The structure of the current full message must be a clone of the original FullSync message structure. However, you must map effective-dated records to a record view that selects only those rows that contain current data.

This example uses the message CUSTOMER_FULLSYNC. The current full message for customer data, CUSTOMER_FULLSYNC, uses the following views that are created as ordering view records.

| Level | TARGET RECORDS | ORDERING VIEW RECORDS |
|---|---|---|
| Level 0 | CUSTOMER | CUSTOMER |
| Level 1 | CUST_ADDR_CNTCTC | CUST_ADDR_CNTCTC |
| Level 1 | CUST_ADDR_SEQ | CUST_ADDR_SEQ |

Copyright © 1992-2006, Oracle. All rights reserved.

| Level | TARGET RECORDS | ORDERING VIEW RECORDS |
|---|---|---|
| Level 2 | CUST_ADDRESS (effective-dated) | CUST_ADDR_EF2VW (current effective-dated view) |
| Level 1 | CUST_CNTCT_SEQ | CUST_CNTCT_SEQ |
| Level 2 | CUST_CONTACT (effective-dated) | CUST_CNCT_EF2VW (current effective-dated view) |
| Level 3 | CUST_CNTCT_CARD (effective-dated) | CUST_CARD_EF_VW (current effective-dated view) |
| Level 3 | CUST_CNTCT_DOC (effective-dated) | CUST_DOC_EF_VW (current effective-dated view) |
| Level 3 | CUST_CNTCT_PHN (effective-dated) | CUST_PHN_EF_VW (current effective-dated view) |
| Level 3 | CUST_CNTCT_TYPE (effective-dated) | CUST_TYPE_EF_VW (current effective-dated view) |

### See Also

*Enterprise PeopleTools 8.48 PeopleBook: PeopleSoft Integration Broker,*"Managing Rowset-Based Messages"

## Creating the Service Operation

To complete a full data publish of effective dated data, you must create a service operation. When you create a service operation:

**Note.** The name of the service operation must end with *_EFF* suffix to be effective dated.

- Specify the effective dated messages to publish.
- Specify a service operation queue for each message.
- Specify the directionality or routing of the integration.
- Define an OnRouteSend service operation handler, which contains the processing logic.

### See Also

*Enterprise PeopleTools 8.48 PeopleBook: PeopleSoft Integration Broker,*"Managing Service Operations"

## Defining the Node and Target Connector

Create a full message definition that contains the necessary records in the publishing system. You can also set up message routing by using OnRouteTo PeopleCode.

 Copyright © 1992-2006, Oracle. All rights reserved.

**Note.** Remember to insert the message version first, otherwise you can't add the tables that compose the message.

Begin by setting up the node and the transaction and connector details by using the Integration Profile setup function of PeopleSoft Integration Broker. To set up the node:

1. Create a node.

   See *Enterprise PeopleTools 8.48 PeopleBook: PeopleSoft Integration Broker, "Adding and Configuring Nodes"*

2. Set up the connector.

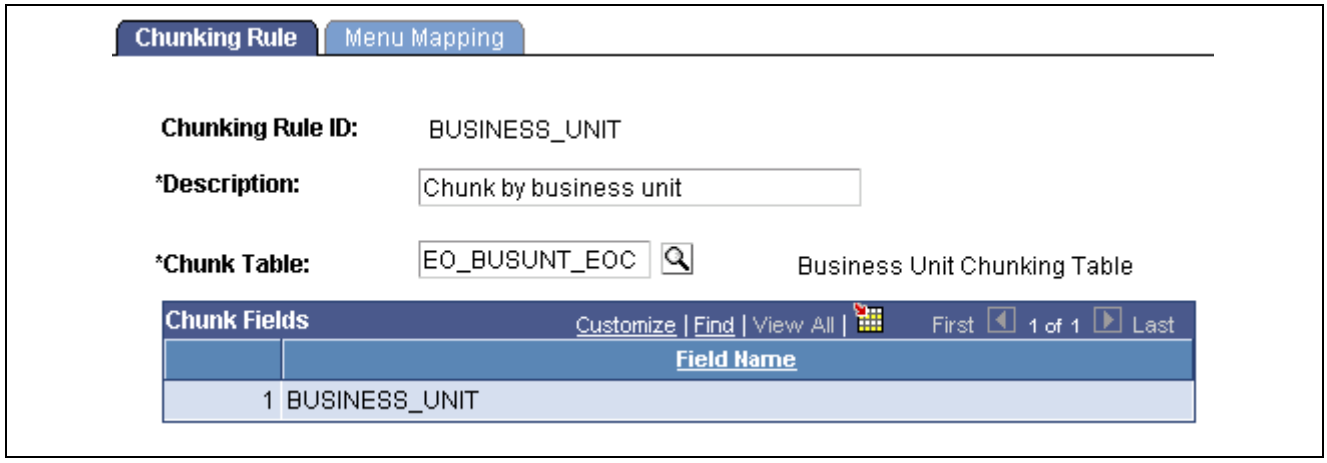
   The default target is PSHTTP, but you can instead provide the HTTP address for another target.

   See *Enterprise PeopleTools 8.48 PeopleBook: PeopleSoft Integration Broker,* "Managing Integration Gateways"

## Defining Chunking Rules and Ordering Views

In message chunking, all data within the message contains the same break field. (For example, if the break field is business unit, all transactions in the message are for the same business unit.)

The following message is chunked on setID by using the EO_SETID_EOC table:




Chunking Rule page

To ensure that the message publishes when you use chunking rules:

1. Add subscribing nodes to the chunking node table.
2. In PeopleSoft Application Designer, add OnRouteSend PeopleCode to return a list of subscribing nodes.

## Creating Publish Rule Definitions

For creating the current full message, the publish rule defines these options:

- Message header and trailer creation.
- Chunking rules.
- Ordering views.

Create publish rule definitions for each current full message definition. Considerations are:

Copyright © 1992-2006, Oracle. All rights reserved.

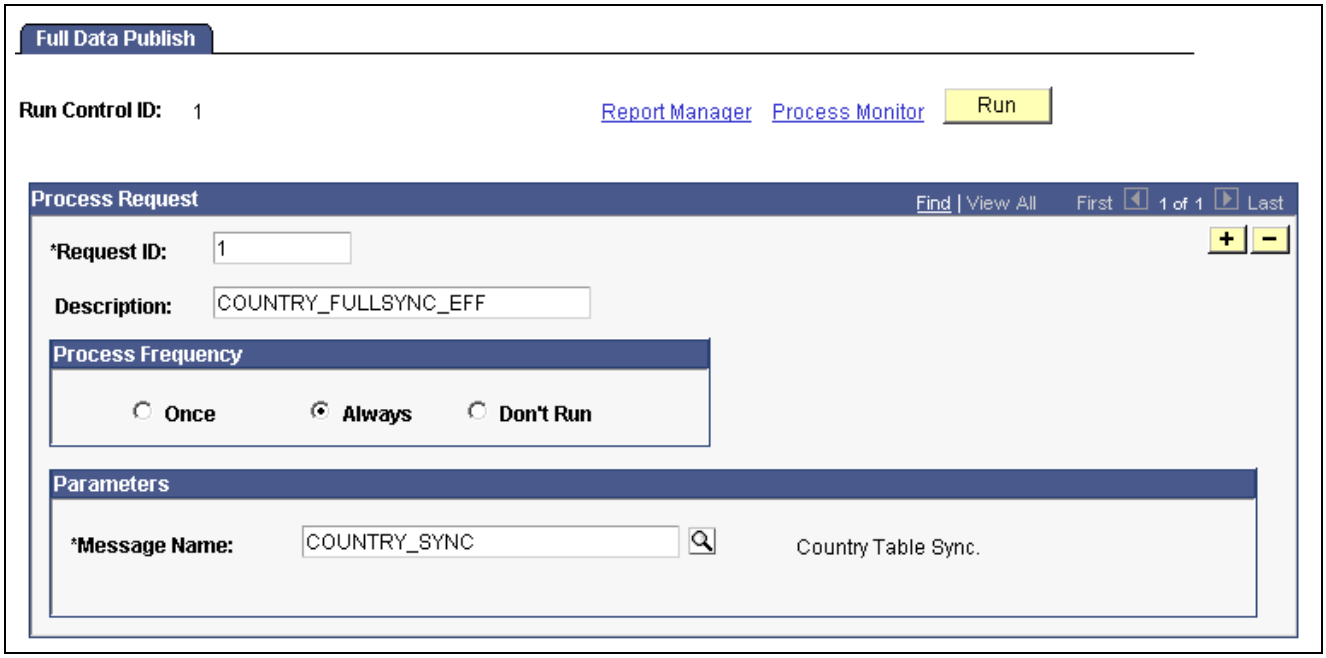
- Specify only target records that are effective-dated.
- Select only the current effective row to list the ordering view record that should be used as an override when publishing the message.
- The Effective Date Publish utility makes a logic pass through the data for each publish rule definition.

  You can use this logic to order and chunk the data differently for each subscriber.

**Note.** When chunking a message, you must provide an ordering view for each record that includes the chunking fields. The fields in this view must appear in the same order as the primary keys, followed by any other keys that are needed for that record. If you override the normal key structure of the message records, you must provide the ordering views for each record to guarantee that the message reconstructs with the correct chunking, parent, or child key relationships.

## Creating Run Controls for the Full Data Publish Program

Access the Full Data Publish page.




Full Data Publish page

**Request ID** Enter request IDs to group the Description, Process Frequency, and Message Name parameters under one unique process request. A single run control ID can encompass multiple request IDs.

**Parameters** Select the name of the message to publish.

The PeopleSoft system adds a run control for the currently effective FullSync message that is chunked by the setID CUSTOMER_FULLSYNC_EFF_SETID.

**Note.** If you insert a new row, the same run control component can publish more than one message, so you can produce both the full message and the current effective-dated full message from the same PeopleSoft Process Scheduler run.


Copyright © 1992-2006, Oracle. All rights reserved.

**Note.** You must set up the run control parameters to start the Full Data Publish program.

### Performing Full Table Replication

After you click Run on the Full Data Publish page, you can perform a table replication publish from the following screen:

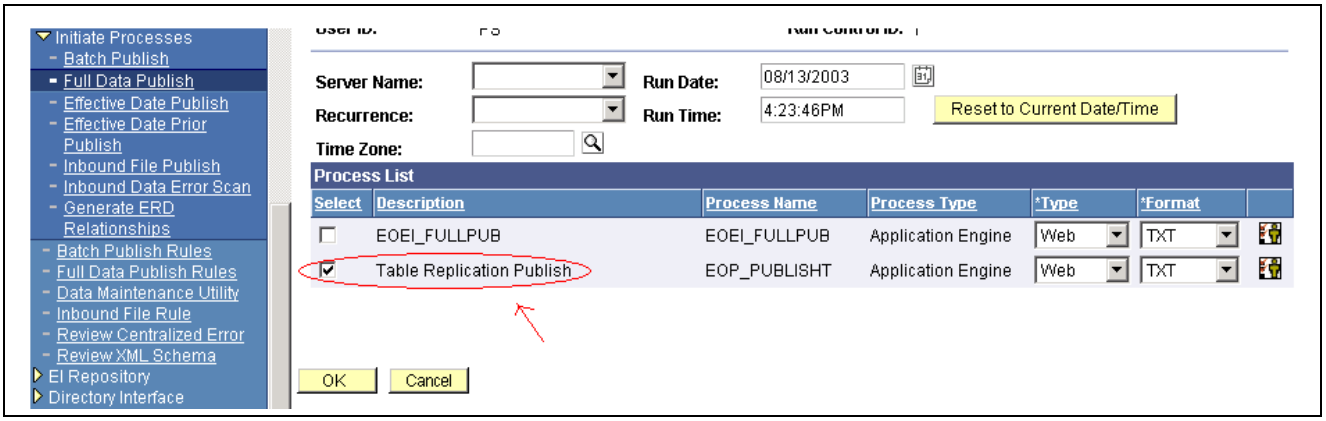



Table Replication Publish checkbox

To perform a full table replication, click the checkbox shown.

## Defining Routing

To define the message routing, you must create an OnRouteSend service operation handler and define the handler in the service operation.

**Note.** Perform this step only if you're chunking a message

### Example

You want to route the customer message to the nodes that are defined in the SetID/Nodes page within the Publish Setup component. Add the following PeopleCode to the OnRouteSend handler for the service operation:

```
Declare Function GETSETIDNODES PeopleCode FUNCLIB_EOEIP.PUBLISH_ROUTE_PC
     Field Formula;
Local Message &MSG;
/* Call Function that looks at Setid of first transaction in the message
 and returns a list of subscribing nodes to route the message */
&MSG = GetMessage();
GETSETIDNODES(&MSG, %Date);
```

# Publishing Incremental Messages of Current Effective Data

This section discusses how to:

- Create a service operation for publishing incremental messages of current effective data..

Copyright © 1992-2006, Oracle. All rights reserved.

- Create subscription processes that open the generic effective-dated delay function.

For incremental messages, use subscription PeopleCode to copy current effective rows to a current incremental message for immediate publication, to strip out historic data, and to store future effective rows in a delay table. A regularly scheduled Application Engine program uses the delay table data as a trigger to publish future data when that data becomes effective.

## Creating Service Operations for Publishing Incremental Messages of Current Effective Data

You must create a service operation and associate an incremental message definition to it.

**Note.** The name of the service operation must end with *_EFF* suffix to be effective dated.

The incremental message definition must contain the necessary records in the publishing system. Specify the message version first, otherwise you can't add the tables that compose the message. When the system requests a message channel, enter the queue definition that you previously selected.

## Creating Subscription Processes That Open the Generic Effective-Dated Delay Function

The PeopleSoft system includes a function called PROCESS_EFFDT_MSG. This function reads through an incremental message and processes the past, current, future, or non-effective-dated information. PROCESS_EFFDT_MSG resides within the record FUNCLIB_EOEIP, in the EFFDT_MSG_PC field.

To create a subscription process that opens the generic effective-dated Delay function:

1. Open the Process_Effdt_Msg generic function stored in record FUNCLIB_EOEIP.
2. Pass it the name of the current effective-dated incremental message.
3. Indicate whether only rows with *Active* effective status should be selected as the current effective-dated rows.

   If the &ACTIVE_EFFSTATUS parameter passed in is set to *False,* the current effective-dated row (whether active or inactive) is selected.
4. Pass the parameter set to True if only active effective-dated rows should be sent to the other system.

   **Note.** The standard setting for the &ACTIVE_EFFSTATUS parameter is *False.*

# Publishing Effective-Dated Rows from the Delay Table

This section provides an overview of the publish of effective-dated rows from the delay table and discusses how to run the Effective Date Publish utility.

## Understanding Effective-Dated Row Publishing

Publishing effective-dated rows from a delay table requires:

- A future-dated entry in the delay table.
- The process request page.
- An Application Engine utility program that publishes future-dated information when it becomes current.

 Copyright © 1992-2006, Oracle. All rights reserved.

The Application Engine Effective Date Publish utility publishes effective-dated rows from a delay table by:

1. Retrieving from the delay table any entries that are effective within a date range.
2. Using the key strings from the delay table and record information for the current message to read the original application tables and retrieve the most current effective rows.
3. Publishing those rows to the current incremental message.

   The third-party application subscribes to the current incremental messages.

The PeopleSoft system allows for an end date range on the Effective Date Publish utility if the utility was not run for one day. The end date range enables the program to run on the next date.

You can run the Effective Date Publish utility multiple times during the day, and it deletes the information from the delay queue when the future effective date data becomes current and the message for that data has been published. The Effective Date Publish utility retrieves only the latest delay table information since the prior run.

If the Effective Date Publish utility is invoked after not running for a period of time, it retrieves only the current row and publishes that as the active record. The presumption is that the subscribers want only the most current database information that is published.

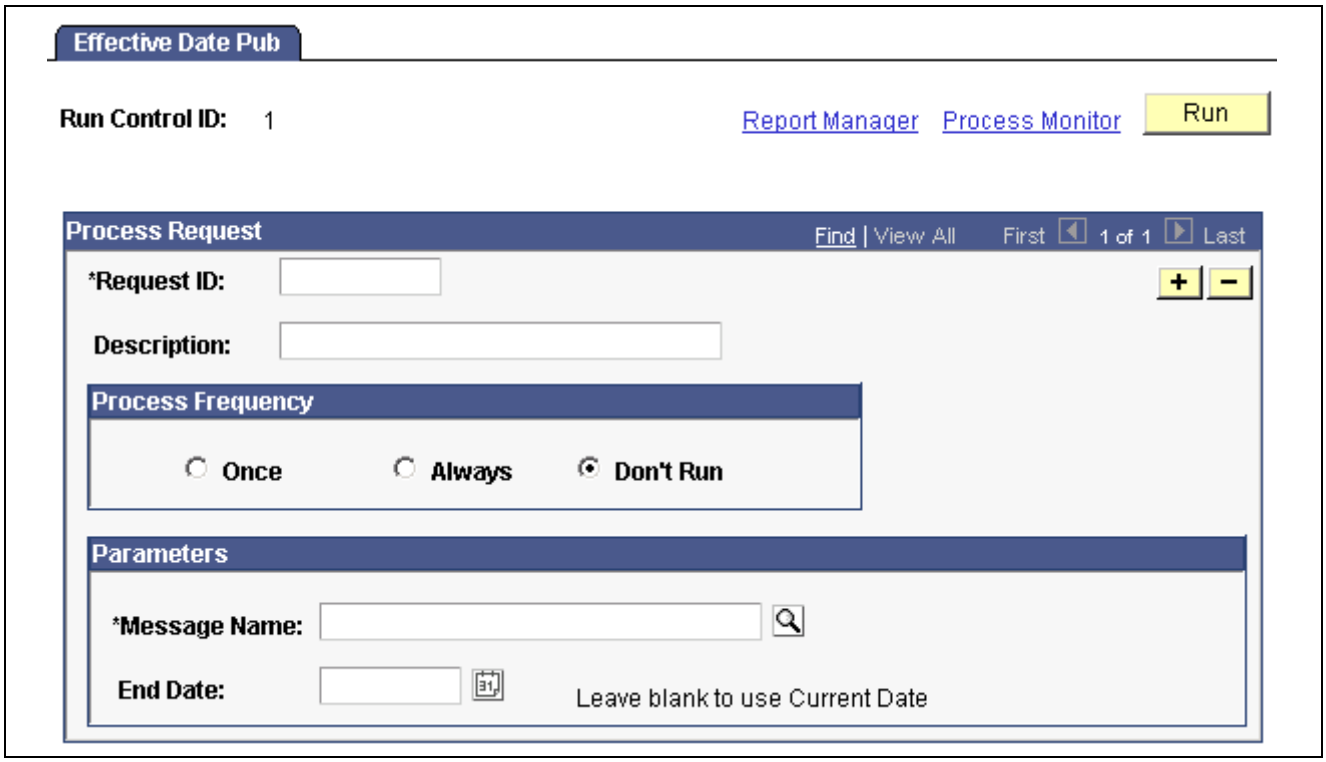
## Page Used to Run the Effective Date Publish Utility

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| Effective Date Pub (effective date publish) | EO_EFFDATAPUB | Enterprise Components, Integration Definitions, Initiate Processes, Effective Date Publish | Run the Effective Date Publish utility. |

## Running the Effective Date Publish Utility

Access the Effective Date Pub page.

Copyright © 1992-2006, Oracle. All rights reserved. 




Effective Date Pub page

**Message Name** Select the current incremental message to publish.

**End Date** Select the highest effective date to process from the delay table.

**Run** Click to run this request.

The Application Engine program uses the trigger records in the delay table and the end date parameter from the run control component to publish a current effective incremental message that contains all future-dated rows that are effective. This ensures that third-party systems that cannot manage future-dated records always receive currently active data on that data's effective date, even if that information was previously updated on the PeopleSoft system.


Copyright © 1992-2006, Oracle. All rights reserved.

# CHAPTER 21

# Using the Flat File Utility

This chapter provides an overview of the Flat File utility and discusses how to:

- Process inbound flat files.
- Initiate file processing.
- Test inbound flat file processing.

## Understanding the Flat File Utility

When external systems send flat files to you for inbound transactions, you must develop complementary processes to translate incoming files into messages or translate outbound messages into files.

The flow for inbound file processing by using the Flat File utility is:

1. The utility receives a flat file in the form of a file layout object from an external system.

   The flat file consists of either:

   - The relevant data.
   - An index file that contains pointers to the data.

     Each index file lists the names of a set of data files to be processed. These files contain the application data, which is in one of the following formats: fixed record, Comma Separated Values (CSV), or XML.

2. The utility reads the file that is submitted for processing:
   - If the file is an index file, the Flat File utility loads the list of data files that are associated with each index file to be processed into a parameter table.
   - If it is a single data file, the utility inserts the single data file into a parameter table.

   **Note.** If additional fields in the file layout are not in the message definition, the additional fields are ignored during the copying of the flat file data to the message and are not included in the message.

3. The utility loops through the list of data files to be processed and reads each data file.
4. The utility copies the row sets of the data files into the message.
5. The utility publishes the message.
6. The subscribing systems receive the message and initiate normal inbound data processing.

Copyright © 1992-2006, Oracle. All rights reserved.

# Processing Inbound Flat Files

You use the file layout definition to read and write from flat files.

To process inbound flat files:

1. Determine the necessary format of the inbound data.

   If there's an industry standard, use it for your file definition.

   If there's no industry standard, create a file layout object that mirrors your message object.
2. Identify the inbound process and its standard message.
3. Analyze the vendor's file structure and compare it to the standard message.

   Questions to answer include:

   - Can you use an existing message, or do you need to create a new one?
   - Can the customer conform to an existing integration point, or do you need to create one (along with corresponding subscription PeopleCode)?
4. Create the message definition.
5. Create a file layout definition with the same structure as the message definition to support the vendor file format.

   The hierarchical structure of the data in the File Layout Definition must match that of the message definition. For example, suppose a message has three levels: level zero, containing Record A, level one, containing records B and C, and level two, containing record D.

   All file layouts that are associated with this message must also have record A in level zero, record B and C in level one, and record D in level two.

   **Note.** The file layout does not need to contain the exact same fields as the message definition.

   For every record in your file layout, add a new file field, AUDIT_ACTN, as the first field in the record (except when the field already exists in the application table).

   You can associate more than one file layout to a single message. For example, vendor A may have a different number of fields than vendor B, so you may have two file layouts: one for A and one for B.

   Specify the file ID uniquely to include a row in a file, which is necessary in mapping the data to its proper record. Include start and end points when dealing with more than one record in a file layout.

   **Note.** Each record in the file layout has a file record ID attribute. Do not confuse this with the file layout ID. The file layout ID determines whether a new layout is encountered for multiple file layout processing.

   When you subscribe to the message and normal inbound data processing begins, you can invoke the SetDefault PeopleCode function to set the default values for fields that were not present in the input file.
6. Update or create the inbound file rule pages.
7. Create subscription PeopleCode in PeopleSoft Application Designer to process the message.

   Have the standard inbound process subscribe to and process the message normally. The standard message definition should have a subscription process that initiates the normal inbound processing for that object to which you hook your application logic to process the file data.

 Copyright © 1992-2006, Oracle. All rights reserved.

8. Test the inbound flat file processing.

**Note.** You can process multiple inbound flat files at one time. By specifying an inbound index file as part of the Flat File utility parameters, the system reads all input files within the index file and uses the associated file layout object and message to convert the data. Similarly, specify a wildcard in the filename in the inbound file rule component, but make sure that all files that meet the wildcard criteria correspond to the file layout and message mapping that are defined.

# Initiating File Processing

This section discusses how to:

- Set up inbound flat file processing.
- Initiate inbound flat file processing.

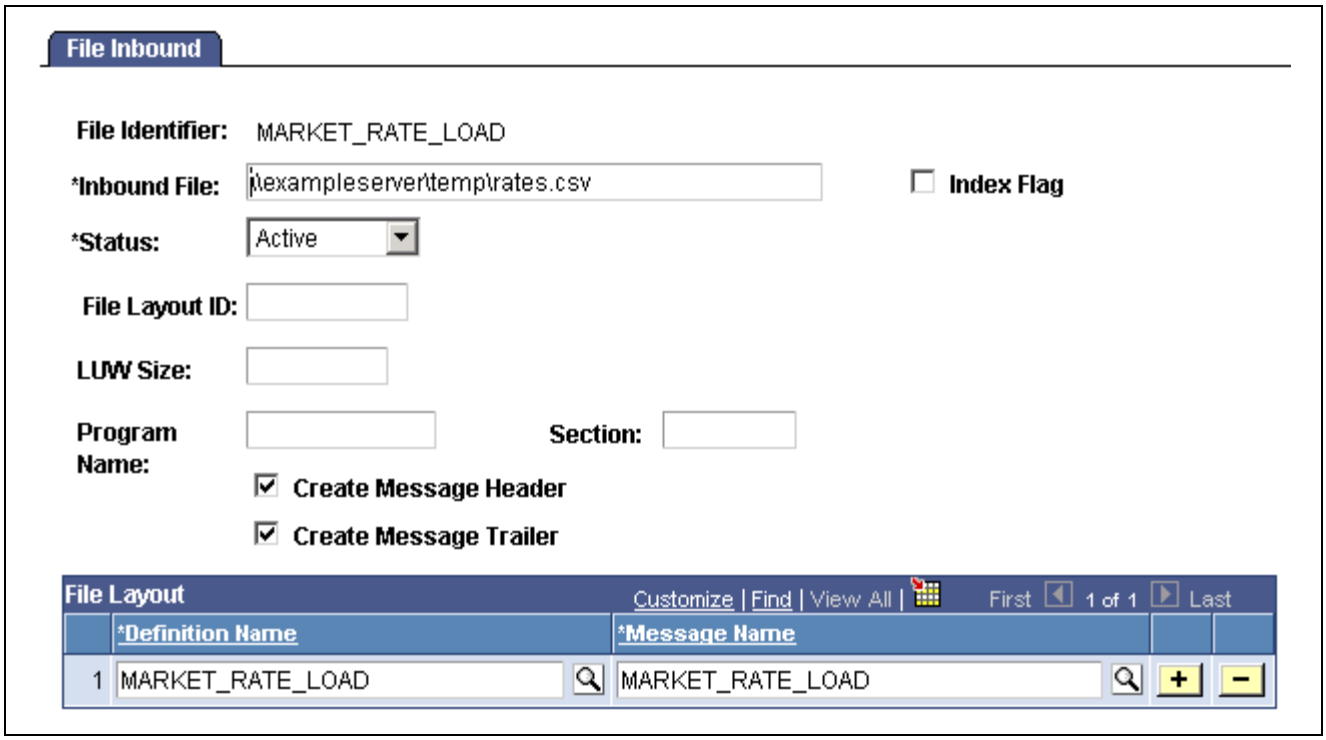
## Pages Used to Initiate File Processing

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| File Inbound | EO_FILE_INBOUND | Enterprise Components, Integration Definitions, Inbound File Rule | Set up inbound flat file processing. |
| Inbound File | EO_FILETOMSG | Enterprise Components, Integration Definitions, Initiate Processes, Inbound File Publish | Initiate inbound flat file processing. This file-to-message processing function reads the file rowset and publishes it as a message. |

## Setting Up Inbound Flat File Processing

Access the File Inbound page.

Copyright © 1992-2006, Oracle. All rights reserved. 




File Inbound page

**File Identifier** Displays the inbound file that you are associating with the rule.

**Inbound File** Enter the index file name or the data file name. Specify the full path information. The PeopleCode program uses the *%filepath_absolute* variable when opening the file.

**Index Flag** Select to distinguish between the index and the data file.

**Status** Select whether this inbound file rule is *Active* or *Inactive.* The default value is *Inactive.*

**File Layout ID** Select a layout to associate with the file.

**LUWSize** (logical unit of work size) Enter the number of level zero rows that are in each message, to limit the message size. The output message is normally determined by the MaxMessageSize system parameter.

**Program Name** and **Section** Select a PeopleSoft Application Engine program and section to invoke when the utility finishes processing data.

**Create Message Header** Select to create a header message. Use the header message as a trigger in the subscription process to initialize tables before receiving the data messages. Default value is selected.

**Create Message Trailer** Select to create a trailer message. Use the trailer message as a trigger in the subscription process to indicate that all the data messages have been received. Default value is selected.


Copyright © 1992-2006, Oracle. All rights reserved.

### File Layout

**Definition Name** and **Message Name**

If the File Layout ID field is blank, this field should contain only one entry.

If the File Layout ID field is not blank, this scroll area must contain an entry for each file layout definition name that is specified in the inbound file.

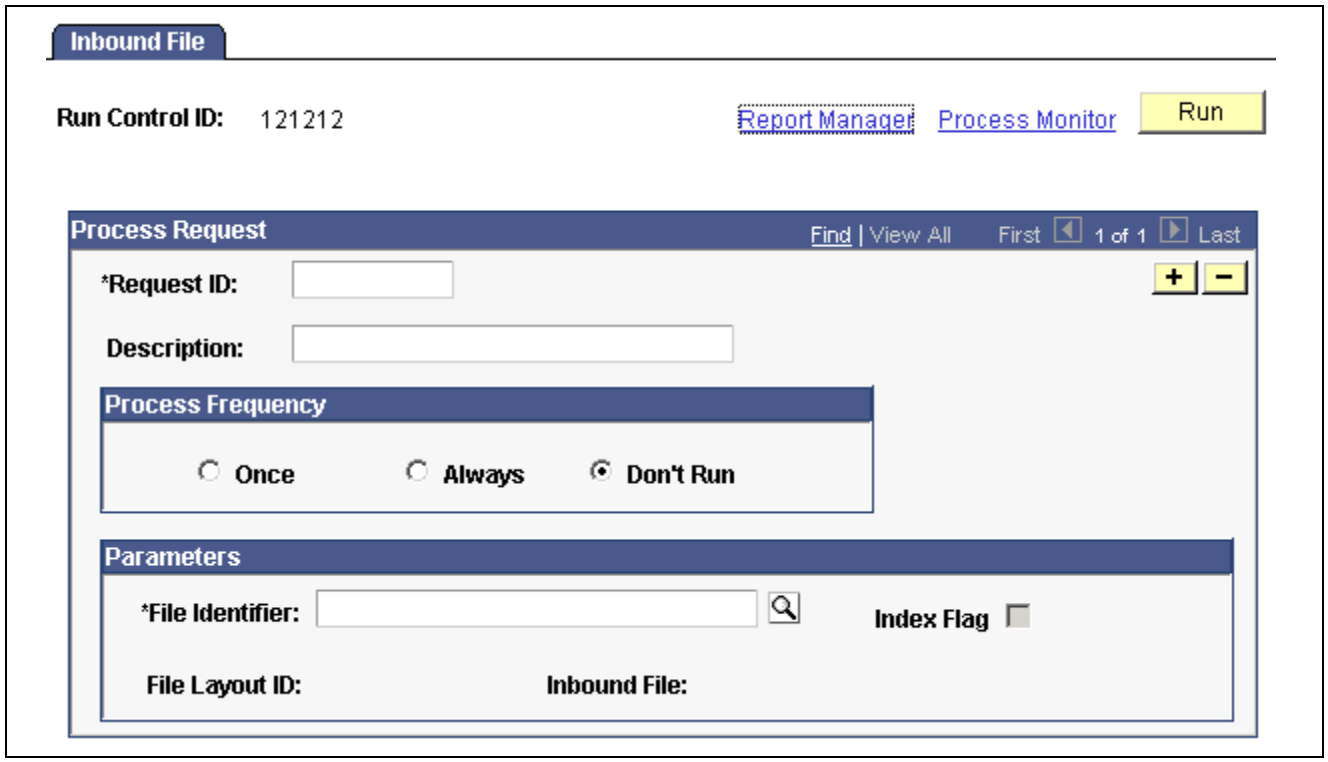
---

**Note.** Use the wildcards * and ? for the file name but not for the directory path. The file layout and message mapping must be valid for all files that meet the wildcard criteria.

---

## Initiating Inbound Flat File Processing

Access the Inbound File page.




Inbound File page

### Parameters

**File Identifier** Select or enter the name of the file identifier that you set up in the File Inbound page. The file identifier is tied to the publish rules.

**Run** Click to run this request.

### Publishing a New Message

The Inbound File page runs an Application Engine process that initiates the file-to-message processing. The file-to-message processing function reads the file rowset and publishes it as a message.

If an index file exists when the inbound conversion process runs, the Application Engine program loads the list of files to be converted into a parameter table and completes a commit. The Application Engine program uses the list of files within the parameter table to restart the processing if a particular flat file fails. If a single data file is provided, then the rowset processing immediately begins.

Copyright © 1992-2006, Oracle. All rights reserved.

The file publish process goes through each of the rowsets of the file layout and copies them into the message row sets.

If the audit action (AUDIT_ACTN) exists in the file, it is copied to the PSCAMA record. If the audit action does not exist in the file, the publishing process uses the default value that is specified in the file layout field property.

The Flat File utility publishes a new message when one of the following exists:

- Maximum message size is exceeded.
- Logical unit of work publish size is reached.
- A new file layout is detected.
- End of file is reached.

The Application Engine program completes a commit every time a message is published from a file. After conversion, the flat file remains in the parameter table with a status of *Processed.*

---

**Note.** The file layout should exactly match the message layout (excluding the PSCAMA record) and should use the same character set as that used by the file: either American National Standards Institute or Unicode.

---

# Testing Inbound Flat File Processing

To test inbound files:

1. Create a sample flat file, or ask the third-party vendor for a sample flat file.
2. Launch the Flat File utility.
    a. Through the browser, sign in to PeopleSoft Internet Architecture.
    b. Select Enterprise Components, Management, Inbound File Rule.
3. Run the Application Engine program to convert the sample flat file to a message by running Message Monitor.

    Use Message Monitor to ensure the inbound file processing created a publish message that contains the sample flat file data.

    a. Verify that the standard inbound subscription process received the message and processed it into the application tables.
    b. Determine whether the values become the inherited values (if you used the inherited value feature in file layout).
    c. Validate that the production or staging tables loaded with the correct field values.

        For production tables, look in the PeopleSoft application pages.

        For staging tables, use either the PeopleSoft application pages or run a query by using PeopleSoft Query.
    d. Ensure that the date formats conform.

 Copyright © 1992-2006, Oracle. All rights reserved.

# CHAPTER 22

# Using the XML Schema Utility

This chapter provides an overview of the XML Schema utility and discusses using the XML Schema utility.

**Note.** The XML Schema Utility has been superseded by PeopleTools Message Schema Builder that is a part of PeopleSoft Integration Broker.

## Understanding the XML Schema Utility

PeopleSoft Open Integration Framework enables near real-time messaging and transactions by using a format that is based on XML to convey information between diverse applications in a standard way. To take advantage of this standardization, you must obtain clear XML definitions (schemas) for each application message, component interface, or business interlink.

The XML Schema utility provides the following features:

- Output options for the XML Schema utility, document type definition (DTD), or BizTalk definition for all application messages.
- The ability to create an XML definition for a single object, for all of the objects, or for all of the objects by a specific owner.
- A single flat file for each XML definition that is written to your system's %TEMP directory (when you use the Microsoft Windows client) or the server's common access file directory (when you use PeopleSoft Internet Architecture).
- An application foundation for future standards of XML definitions.

## Generating the XML Schema

This section discusses how to generate the XML Schema.

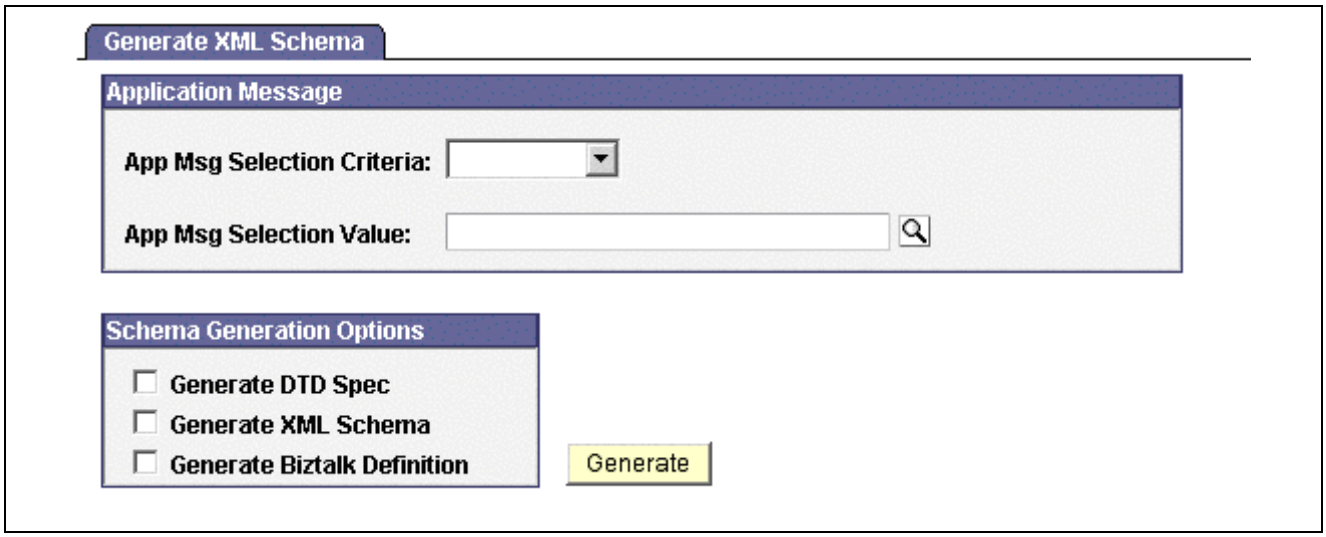
### Page Used to Generate XML Schema

| Page Name | Object Name | Navigation | Usage |
|---|---|---|---|
| XML Schema | EO_GEN_XML_DATA | Enterprise Components, Integration Definitions, Review XML Schema | Generate DTDs, XML schemas, and BizTalk definitions. |

### Generating the XML Schema

Access the Generate XML Schema page.

Copyright © 1992-2006, Oracle. All rights reserved.




Generate XML Schema page

| | |
|---|---|
| **App Msg Selection Criteria** (application message selection criteria) | Select application selection message criteria. Values are *All Msg* (all messages), *Channel, Owner ID,* and *Single Msg* (single message). Depending on the selection criteria, you can enter the selection value (if already known) or search for the value. |
| **Generate DTD Spec** | Select for DTD spec output format. |
| **Generate XML Schema** | Select for XML schema output format. |
| **Generate Biztalk Definition** | Select for BizTalk definition output format. |
| **Generate** | Click to generate the selected output formats. |

The utility queries the relevant PeopleTools tables to generate the selected types of XML schemas and writes the results to the server's file directory or your system's Temp directory, depending on the client you use.

To produce an XML schema, DTD, or BizTalk definition:

1. Define the selection criteria for application messages, component interfaces, and business interlinks.
2. Select XML schemas, DTDs, or BizTalk definitions for application message, component interface, and business interlink objects.

   PeopleCode sends a query to the PeopleTools tables to create the selected types of XML definitions.

   The XML Schema utility then writes the definitions to the file directory as specified by the PS_SERVDIR environment variable when using PeopleSoft Internet Architecture or the %TEMP directory of your system when using the Microsoft Windows client.

# Interpreting Sample Output

The following code shows three samples of output for the same application message (in this case, MARKET_RATE_TYPE_FULLSYNC) in BizTalk, DTD, and XML schema formats.

## Example: BizTalk

The following code shows MARKET_RATE_TYPE_FULLSYNC in BizTalk format:

```
<BizTalk xmlns="urn:schemas-biztalk-org:BizTalk/biztalk-0.81.xml">
```


Copyright © 1992-2006, Oracle. All rights reserved.

```
<Body>
<xsd:schema xmlns:xsd="http://www.w3.org/2000/10/XMLSchema">
<xsd:element name="MARKET_RATE_TYPE_FULLSYNC"
  type="MARKET_RATE_TYPE_FULLSYNCType"/>

<xsd:complexType name="MARKET_RATE_TYPE_FULLSYNCType">
<xsd:sequence>
   <xsd:element name="FieldTypes" type="FieldTypesType"/>
   <xsd:element name="MsgData" type="MsgDataType"/>
</xsd:sequence>
</xsd:complexType>

<xsd:complexType name="FieldTypesType">
<xsd:sequence>
   <xsd:element name="RT_TYPE_TBL" type="FieldTypesRT_TYPE_TBLType"/>
   <xsd:element name="PSCAMA" type="PSCAMA"/>
</xsd:sequence>
</xsd:complexType>

<xsd:complexType name="FieldTypesRT_TYPE_TBLType">
<xsd:sequence>
   <xsd:element name="RT_TYPE" type="FieldTypesFieldType"/>
   <xsd:element name="DESCR" type="FieldTypesFieldType"/>
   <xsd:element name="DESCRSHORT" type="FieldTypesFieldType"/>
</xsd:sequence>
   <xsd:attribute name="class" type="xsd:string" use="required" value="R"/>
</xsd:complexType>

<xsd:complexType name="PSCAMA">
   <xsd:sequence>
   <xsd:element name="LANGUAGE_CD" type="LANGUAGE_CDType" minOccurs="0"
     maxOccurs "1"/>
   <xsd:element name="AUDIT_ACTN" type="AUDIT_ACTNType"/>
   <xsd:element name="BASE_LANGUAGE_CD" type="BASE_LANGUAGE_CDType"
     minOccurs="0"maxOccurs="1"/>
   <xsd:element name="MSG_SEQ_FLG" type="MSG_SEQ_FLGType" minOccurs="0"
     maxOccurs= "1"/>
   <xsd:element name="PROCESS_INSTANCE" type="PROCESS_INSTANCEType"
     minOccurs="0" maxOccurs="1"/>
   <xsd:element name="PUBLISH_RULE_ID" type="PUBLISH_RULE_IDType"
     minOccurs="0" maxOccurs="1"/>
   <xsd:element name="MSGNODENAME" type="MSGNODENAMEType" minOccurs="0"
     maxOccurs= "1"/>
   </xsd:sequence>
   <xsd:attribute name="class" type="xsd:string" use="required" value="R"/>
</xsd:complexType>

<xsd:complexType name="LANGUAGE_CDType" >
   <xsd:attribute name="type" type="fieldtypes" use="optional"/>
</xsd:complexType>
```

Copyright © 1992-2006, Oracle. All rights reserved.

```
<xsd:complexType name="AUDIT_ACTNType" >
   <xsd:attribute name="type" type="fieldtypes" use="optional"/>
</xsd:complexType>

<xsd:complexType name="BASE_LANGUAGE_CDType"  >
   <xsd:attribute name="type" type="fieldtypes" use="optional"/>
</xsd:complexType>

<xsd:complexType name="MSG_SEQ_FLGType">
   <xsd:attribute name="type" type="fieldtypes" use="optional"/>
</xsd:complexType>

<xsd:complexType name="PROCESS_INSTANCEType">
   <xsd:attribute name="type" type="fieldtypes" use="optional"/>
</xsd:complexType>

<xsd:complexType name="PUBLISH_RULE_IDType">
   <xsd:attribute name="type" type="fieldtypes" use="optional"/>
</xsd:complexType>

<xsd:complexType name="MSGNODENAMEType">
   <xsd:attribute name="type" type="fieldtypes" use="optional"/>
</xsd:complexType>


<xsd:complexType name="MsgDataType">
      <xsd:sequence>
   <xsd:element name="Transaction">
      <xsd:complexType>
      <xsd:sequence>
         <xsd:element name="RT_TYPE_TBL" type="MsgDataRT_TYPE_TBLType"/>
         <xsd:element name="PSCAMA" type="PSCAMA"/>
      </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
      </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="MsgDataRT_TYPE_TBLType">
<xsd:sequence>
   <xsd:element name="RT_TYPE">
<xsd:simpleType>
<xsd:restriction base="xsd:string" >
   <xsd:pattern value="[A-Z]{1-5}" />
</xsd:restriction>
</xsd:simpleType>
</xsd:element>
   <xsd:element name="DESCR">
<xsd:simpleType>
```

 Copyright © 1992-2006, Oracle. All rights reserved.

```
<xsd:restriction base="xsd:string">
   <xsd:pattern value=".{1-30}" />
</xsd:restriction>
</xsd:simpleType>
</xsd:element>
   <xsd:element name="DESCRSHORT">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
   <xsd:pattern value=".{1-10}" />
</xsd:restriction>
</xsd:simpleType>
</xsd:element>
</xsd:sequence>
   <xsd:attribute name="class" type="xsd:string" use="required" value="R"/>
</xsd:complexType>

<xsd:complexType name="FieldTypesFieldType">
   <xsd:attribute name="type" type="fieldtypes"/>
</xsd:complexType>

<xsd:simpleType name="fieldtypes">
<xsd:restriction base="xsd:string">
   <xsd:enumeration value="CHAR"/>
   <xsd:enumeration value="NUMBER"/>
   <xsd:enumeration value="DATE"/>
   <xsd:enumeration value="DATETIME"/>
   <xsd:enumeration value="TIME"/>
</xsd:restriction>
</xsd:simpleType>

</xsd:schema>
</Body>
</BizTalk>
```

## Example: DTD

The following code shows MARKET_RATE_TYPE_FULLSYNC in DTD format:

```
<!ELEMENT MARKET_RATE_TYPE_FULLSYNC (FieldTypes, MsgData)>

   <!ENTITY % recordtypes "class (R | SR) #REQUIRED" >
   <!ENTITY % fieldtypes "type (CHAR | NUMBER | DATE | TIME | DATETIME) #IMPLIED" >

<!ELEMENT FieldTypes (RT_TYPE_TBL, PSCAMA)>

<!ELEMENT PSCAMA (LANGUAGE_CD?, AUDIT_ACTN, BASE_LANGUAGE_CD?, MSG_SEQ_FLG?,

 PROCESS_INSTANCE?, PUBLISH_RULE_ID?, MSGNODENAME?)>
   <!ATTLIST PSCAMA class (R | SR) #REQUIRED>
   <!ELEMENT LANGUAGE_CD (#PCDATA)>
```

Copyright © 1992-2006, Oracle. All rights reserved.

```
      <!ATTLIST LANGUAGE_CD type (CHAR | NUMBER | DATE | TIME | DATETIME) #IMPLIED>
   <!ELEMENT AUDIT_ACTN (#PCDATA)>
      <!ATTLIST AUDIT_ACTN type (CHAR | NUMBER | DATE | TIME | DATETIME) #IMPLIED>
   <!ELEMENT BASE_LANGUAGE_CD (#PCDATA)>
      <!ATTLIST BASE_LANGUAGE_CD type (CHAR | NUMBER | DATE | TIME |
        DATETIME) #IMPLIED>
   <!ELEMENT MSG_SEQ_FLG (#PCDATA)>
      <!ATTLIST MSG_SEQ_FLG type (CHAR | NUMBER | DATE | TIME | DATETIME) #IMPLIED>
   <!ELEMENT PROCESS_INSTANCE (#PCDATA)>
      <!ATTLIST PROCESS_INSTANCE type (CHAR | NUMBER | DATE | TIME |
        DATETIME) #IMPLIED>
   <!ELEMENT PUBLISH_RULE_ID (#PCDATA)>
      <!ATTLIST PUBLISH_RULE_ID type (CHAR | NUMBER | DATE | TIME |
        DATETIME) #IMPLIED>
   <!ELEMENT MSGNODENAME (#PCDATA)>
      <!ATTLIST MSGNODENAME type (CHAR | NUMBER | DATE | TIME | DATETIME) #IMPLIED>
<!ELEMENT RT_TYPE_TBL (RT_TYPE, DESCR, DESCRSHORT)>
   <!ATTLIST RT_TYPE_TBL class (R | SR) #REQUIRED>

<!ELEMENT RT_TYPE (#PCDATA)>
      <!ATTLIST RT_TYPE type (CHAR | NUMBER | DATE | TIME | DATETIME) #IMPLIED>
   <!ELEMENT DESCR (#PCDATA)>
      <!ATTLIST DESCR type (CHAR | NUMBER | DATE | TIME | DATETIME) #IMPLIED>
   <!ELEMENT DESCRSHORT (#PCDATA)>
      <!ATTLIST DESCRSHORT type (CHAR | NUMBER | DATE | TIME | DATETIME) #IMPLIED>

<!ELEMENT MsgData (Transaction)>
<!ELEMENT Transaction (RT_TYPE_TBL, PSCAMA)>
```

## Example: XML Schema

The following code shows MARKET_RATE_TYPE_FULLSYNC in XML schema format:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2000/10/XMLSchema">

<xsd:element name="MARKET_RATE_TYPE_FULLSYNC"
  type="MARKET_RATE_TYPE_FULLSYNCType"/>

<xsd:complexType name="MARKET_RATE_TYPE_FULLSYNCType">
<xsd:sequence>
   <xsd:element name="FieldTypes" type="FieldTypesType"/>
   <xsd:element name="MsgData" type="MsgDataType"/>
</xsd:sequence>
</xsd:complexType>

<xsd:complexType name="FieldTypesType">
<xsd:sequence>
   <xsd:element name="RT_TYPE_TBL" type="FieldTypesRT_TYPE_TBLType"/>
   <xsd:element name="PSCAMA" type="PSCAMA"/>
</xsd:sequence>
```

 Copyright © 1992-2006, Oracle. All rights reserved.

```
</xsd:complexType>

<xsd:complexType name="FieldTypesRT_TYPE_TBLType">
<xsd:sequence>
   <xsd:element name="RT_TYPE" type="FieldTypesFieldType"/>
   <xsd:element name="DESCR" type="FieldTypesFieldType"/>
   <xsd:element name="DESCRSHORT" type="FieldTypesFieldType"/>
</xsd:sequence>
   <xsd:attribute name="class" type="xsd:string" use="required" value="R"/>
</xsd:complexType>

<xsd:complexType name="PSCAMA">
   <xsd:sequence>
   <xsd:element name="LANGUAGE_CD" type="LANGUAGE_CDType" minOccurs="0"
     maxOccurs= "1"/>
   <xsd:element name="AUDIT_ACTN" type="AUDIT_ACTNType"/>
   <xsd:element name="BASE_LANGUAGE_CD" type="BASE_LANGUAGE_CDType"
     minOccurs="0" maxOccurs="1"/>
   <xsd:element name="MSG_SEQ_FLG" type="MSG_SEQ_FLGType" minOccurs="0"
     maxOccurs= "1"/>
   <xsd:element name="PROCESS_INSTANCE" type="PROCESS_INSTANCEType"
     minOccurs="0" maxOccurs="1"/>
   <xsd:element name="PUBLISH_RULE_ID" type="PUBLISH_RULE_IDType"
     minOccurs="0" maxOccurs="1"/>
   <xsd:element name="MSGNODENAME" type="MSGNODENAMEType" minOccurs="0"
     maxOccurs= "1"/>
   </xsd:sequence>
   <xsd:attribute name="class" type="xsd:string" use="required" value="R"/>
</xsd:complexType>

<xsd:complexType name="LANGUAGE_CDType" >
   <xsd:attribute name="type" type="fieldtypes" use="optional"/>
</xsd:complexType>

<xsd:complexType name="AUDIT_ACTNType" >
   <xsd:attribute name="type" type="fieldtypes" use="optional"/>
</xsd:complexType>

<xsd:complexType name="BASE_LANGUAGE_CDType"  >
   <xsd:attribute name="type" type="fieldtypes" use="optional"/>
</xsd:complexType>

<xsd:complexType name="MSG_SEQ_FLGType">
   <xsd:attribute name="type" type="fieldtypes" use="optional"/>
</xsd:complexType>

<xsd:complexType name="PROCESS_INSTANCEType">
   <xsd:attribute name="type" type="fieldtypes" use="optional"/>
</xsd:complexType>
```

Copyright © 1992-2006, Oracle. All rights reserved.

```
<xsd:complexType name="PUBLISH_RULE_IDType">
   <xsd:attribute name="type" type="fieldtypes" use="optional"/>
</xsd:complexType>

<xsd:complexType name="MSGNODENAMEType">
   <xsd:attribute name="type" type="fieldtypes" use="optional"/>
</xsd:complexType>


<xsd:complexType name="MsgDataType">
      <xsd:sequence>
   <xsd:element name="Transaction">
      <xsd:complexType>
      <xsd:sequence>
         <xsd:element name="RT_TYPE_TBL" type="MsgDataRT_TYPE_TBLType"/>
         <xsd:element name="PSCAMA" type="PSCAMA"/>
      </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
      </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="MsgDataRT_TYPE_TBLType">
<xsd:sequence>
   <xsd:element name="RT_TYPE">
<xsd:simpleType>
<xsd:restriction base="xsd:string" >
   <xsd:pattern value="[A-Z]{1-5}" />
</xsd:restriction>
</xsd:simpleType>
</xsd:element>
   <xsd:element name="DESCR">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
   <xsd:pattern value=".{1-30}" />
</xsd:restriction>
</xsd:simpleType>
</xsd:element>
   <xsd:element name="DESCRSHORT">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
   <xsd:pattern value=".{1-10}" />
</xsd:restriction>
</xsd:simpleType>
</xsd:element>
</xsd:sequence>
   <xsd:attribute name="class" type="xsd:string" use="required" value="R"/>
</xsd:complexType>

<xsd:complexType name="FieldTypesFieldType">
```

 Copyright © 1992-2006, Oracle. All rights reserved.

```
   <xsd:attribute name="type" type="fieldtypes"/>
</xsd:complexType>

<xsd:simpleType name="fieldtypes">
<xsd:restriction base="xsd:string">
   <xsd:enumeration value="CHAR"/>
   <xsd:enumeration value="NUMBER"/>
   <xsd:enumeration value="DATE"/>
   <xsd:enumeration value="DATETIME"/>
   <xsd:enumeration value="TIME"/>
</xsd:restriction>
</xsd:simpleType>

</xsd:schema>
```

Copyright © 1992-2006, Oracle. All rights reserved.

 Copyright © 1992-2006, Oracle. All rights reserved.

# APPENDIX A

# Integration Point Naming Standards

Integration Point names describe the business purposes of integration points and provide a virtual wrapper for packaging one or more integration technology objects.

Integration point names use an approved list of standard PeopleSoft business object nouns and action and event verbs.

## Standard Action and Event Verbs

The following standard verbs are used in the names and descriptions of integration points:

| Verb | Description |
|---|---|
| Acknowledge | Indicates receipt of a processing request. Also conveys the result of the original request. (For example, your application acknowledges a purchase order (PO) when a PO has been issued and the corresponding business application acknowledges the receipt of the PO and responds with an acceptance or counter offer.) |
| Add | Use when a complete entity has already been constructed and needs to be communicated for the first time to another business application, and when there are business implications beyond what a Sync message would convey. Transactional messages may use Add; setup data should use Sync. |
| Adjust | Clarifies a specific process (for example, adjustment of inventory quantity on hand when neither Add, Change, or Delete conveys the full meaning). |
| Allocate | Clarifies a specific process (for example, allocating costs to different business applications when neither Add, Change, or Delete conveys the full meaning). |
| Approve | Use when an entity passes an approval process and is ready for the next business process step. |

Copyright © 1992-2006, Oracle. All rights reserved.

| Verb | Description |
|---|---|
| Cancel | Use when there are business implications beyond a simple change or delete (for example, canceling a purchase order). |
| Change | Use when an entity is changed based on a business event and the change needs to be communicated to another business application. Encompasses business implications beyond what a Sync message would convey (for example, SalesOrderChange). Transactional messages can use Change; setup data should use Sync. |
| Confirm | Responds to a request from the receiving application to confirm. This function conveys the result of the original request (for example, when an inventory issue must be confirmed in an application based on an event in a warehouse management business application). |
| Create | Use when the processing must initiate the building of the document rather than moving the document from one system to another. |
| Find | Requests a list of items from a business application. The response to this request is List. Equivalent to a search dialog box in which you pass the search criteria over as the message. |
| FullSync | Replicates a complete entity, including all record instances, between business applications to initially seed the receiving application with that data. Use with all full message definitions. |
| Get | Requests a specific data entity from a business application. The response to this request is Show. Differs from Find in that the specific entity's key values are known and its details are being requested, whereas Find checks for existence and returns a list of values that match the Find criteria. |
| Issue | Clarifies a specific process (for example, issue material from inventory in cases where neither Add, Change, or Delete conveys the full meaning). |


Copyright © 1992-2006, Oracle. All rights reserved.

| Verb | Description |
|---|---|
| List | Use when sending a list of multiple data entities in a summary format. The List verb can be used to respond to a Find or Get request, or in a publish scenario, to push information to other applications based on a business event. The results of a List can be used as is, or they can be used to select a specific instance of a document or entity to issue a detail Get request. |
| Load | Initiates the addition of a data entity to another business application where maintenance to the document passes to the receiving application permanently. When the request is passed, the sending application no longer has direct control over the document or entity. |
| Post | Clarifies a specific process (for example, a posted journal entry, in cases where neither Add, Change, or Delete conveys the full meaning). |
| Receive | Clarifies a specific process (for example, when you receive inventory against a PO, in cases where Change is not detailed enough for the business context). |
| Request | Requests specific data from a business application. The requested data should be passed back as a separate message. |
| Show | Sends information about a specific instance of a business entity. Can also be used to respond to a Get request, or in a publish scenario in which it pushes information to other applications based on a business event. |
| Sync | Communicates the need to update master files between business applications. Facilitates application integrity and ease of data entry for the business user by enabling a single point of input. Should contain only incremental messages of Add, Change, or Delete actions to the entity. Normally used for passing setup messages between applications. |

Copyright © 1992-2006, Oracle. All rights reserved.

| Verb | Description |
|---|---|
| Transfer | Clarifies a specific process (for example, in the event of a transfer of material from one inventory location to another, in cases where neither Add, Change, or Delete conveys the full meaning). |
| Update | Clarifies a specific process (for example, in the event of an update of inspection information from one business application to another, in cases where neither Add, Change, or Delete conveys the full meaning). The event is not adding a document or changing fields, but communicating the occurrence of an event as well as the corresponding data that accompanies the event. Transactional data may use Update; setup data should use Sync. |

# Standard Business Object Nouns

Business objects appear in uppercase with underscores between key words.

The following table lists examples of business object names only; it is not intended to list or set conventions.

| Account_Chartfield | Deal | Names_Prefix_Suffix | Project_Category |
|---|---|---|---|
| Action_Reason | Department | Nations_Duevo | Project_Status |
| APE_Industry | Dept_Budget | Occupation_Illness | Project_Team |
| Applicant | Detail_Calendar | Par_Location | Project_Type_Cat_ Link |
| Bank | DirectDeposit | Par_Location_Count | Purchase_Order |
| BOM | Earnings | Payroll | Rating_Model |
| Budget | Expense | Payroll_Paysheet | Regulatory_Region |
| Bus_Unit_FS | Expense_Advance | Pension_Fund | Resource_Category |
| Bus_Unit_GL | Expense_Report | Person_ Accomplishment | Resource_SubCategory |
| Bus_Unit_HR | Expense_Sheet | Person_Competency | Resource_Type |


Copyright © 1992-2006, Oracle. All rights reserved.

| Account_Chartfield | Deal | Names_Prefix_Suffix | Project_Category |
|---|---|---|---|
| Bus_Unit_PC | Fund | Person_Contract_ Belgium | Review_Scale |
| Bus_Unit_PF | Grant | Person_Credit_Card | Salary |
| Carrier | Image | Person_Disability | Salary_Matrix |
| Company | Industry_Inspection | Person_Diversity | Salary_Plan |
| Company_Credit_ Card | InterUnit | Person_Education | Salary_Structure |
| Company_Property | Inventory | Person_Names | SalesOrder |
| Competency | Item | Person_Property | Schedule |
| Consumer | Item_BusUnit | Person_PriorWork | State |
| Consumer_Usage | Item_Rev | Person_Visa_Citizen | State_Name |
| Contract | Item_Vendor | Position | Statute |
| ContractBelgium | Job_Code | Product_Chartfield | Statute_Belgium |
| Country | Journal | Product_Item | Vendor |
| Customer | Labor_Category | Product_Group | Visa_Permit |
| Credit_Card | Labor_Relations | Product_Price | Voucher |
| Credit_Card_ Merchant | Location | Product_UOM | Union |
| Currency | Market_Price | Project | UOM |
| Customer | Market_Rate | Project_Activity | Workforce |

Copyright © 1992-2006, Oracle. All rights reserved.

 Copyright © 1992-2006, Oracle. All rights reserved.

# APPENDIX B

# PeopleSoft Design Patterns

PeopleSoft delivers a group of design patterns with each application. The Design Patterns page lists available design patterns.

## List of Design Patterns

The following table presents brief descriptions of the delivered design patterns:

| Design Pattern Name | Description |
|---|---|
| AE Row By Row Publish (application engine row by row publish) | In this design pattern, the transaction or setup data that you want to send out of PeopleSoft is updated by using an Application Engine program that performs procedural (row-by-row) processing; you want to publish these changes. Generally, messages are used with this design pattern. |
| Batch Publish | Use this design pattern to publish messages from a batch application. The batch application can be a COBOL or Structured Query Report program that takes either a procedural or set-based approach, or it can be an Application Engine set-based program. |
| Batch Subscribe | This design pattern enables you to perform edits against messages in sets. This can be a useful technique for high volume data, including millions of inbound rows. This design pattern is useful when you know that a single message definition may contain multiple instances of a transaction, or when you must reuse an existing batch program. |
| CI Subscribe (component interface subscribe) | This design pattern uses a component interface to edit incoming message data. This enables you to reuse existing business rules when processing data. |

Copyright © 1992-2006, Oracle. All rights reserved.

| Design Pattern Name | Description |
|---|---|
| Component Publish | In this design pattern, the transaction or setup data that you want to send out of PeopleSoft is being updated by using a PeopleSoft component. In this case, the data is already in the component buffer, and the Publish PeopleCode function is used to publish a message. |
| EDI In | This design pattern is for inbound EDI documents.<br>You should only use EDI for existing EDI manager inbound transactions that must be supplied to an EDI partner, and you want to allow subscription to an XML message, or when you need to comply with other industry standards, such as SWIFT, BAI, or HL7, that have an existing EDI manager inbound map, and you want to convert to subscribing to an XML message. |
| EDI Out | This design pattern is for outbound EDI documents.<br>You should only use EDI for existing EDI manager inbound transactions that must be supplied to an EDI partner and you want to allow subscription to an XML message, or you need to comply with other industry standards, such as SWIFT, BAI, or HL7, that have an existing EDI manager inbound map, and you want to convert to publishing to an XML message. |
| Full Table Publish | Use this design pattern to populate an entire copy of a table onto a remote database or legacy system. Generally, full data replication occurs with setup tables, or relatively static, low-volume tables that are keyed by setID. When a copy of a table exists on the remote system, incremental updates can be used. |
| Full Table Subscribe | Use this design pattern to subscribe to messages that contain an entire copy of a table that is published from a remote database or legacy system. Generally, full data replication occurs with setup tables, or relatively static, low-volume tables that are keyed by setID. When a copy of a table exists on the remote system, incremental updates can be used. |
| No Pattern | No design pattern is specified. |
| PeopleCode Subscribe | Use this design pattern to subscribe to messages by using a PeopleCode program when additional processing is required. Use PeopleCode subscription when simple edits or no edits against the inbound data are needed before inserting the data into the application tables or staging tables. |


Copyright © 1992-2006, Oracle. All rights reserved.

| Design Pattern Name | Description |
| --- | --- |
| Sync Reply | In this design pattern, another system initiates a request for information from PeopleSoft and waits for information to be returned. This information must be provided by PeopleSoft in a real-time synchronous mode and in a conversational style of interface before the other system can continue processing. Generally, business interlinks are used to satisfy this type of request. |
| Sync Request | Use this design pattern when a PeopleSoft application must call a third-party vendor's application to request information. This information must be provided in a real-time, synchronous mode. Generally, business interlinks are used to satisfy this type of request. |
| XML Reply | In this design pattern, another system initiates a request for information from PeopleSoft. This information must be provided by PeopleSoft in a real-time synchronous mode and in a conversational style of interface before the other system can continue processing. Generally, XMLDocs are used to satisfy this type of request. |
| XML Request | Use this design pattern when a PeopleSoft application must call a third party vendor's application to request information. This information must be provided in a real-time, synchronous mode. Generally, XMLDocs are used to satisfy this type of request. |

Copyright © 1992-2006, Oracle. All rights reserved.

 Copyright © 1992-2006, Oracle. All rights reserved.

# Glossary of PeopleSoft Enterprise Terms

| | |
|---|---|
| **absence entitlement** | This element defines rules for granting paid time off for valid absences, such as sick time, vacation, and maternity leave. An absence entitlement element defines the entitlement amount, frequency, and entitlement period. |
| **absence take** | This element defines the conditions that must be met before a payee is entitled to take paid time off. |
| **academic career** | In PeopleSoft Enterprise Campus Solutions, all course work that a student undertakes at an academic institution and that is grouped in a single student record. For example, a university that has an undergraduate school, a graduate school, and various professional schools might define several academic careers—an undergraduate career, a graduate career, and separate careers for each professional school (law school, medical school, dental school, and so on). |
| **academic institution** | In PeopleSoft Enterprise Campus Solutions, an entity (such as a university or college) that is independent of other similar entities and that has its own set of rules and business processes. |
| **academic organization** | In PeopleSoft Enterprise Campus Solutions, an entity that is part of the administrative structure within an academic institution. At the lowest level, an academic organization might be an academic department. At the highest level, an academic organization can represent a division. |
| **academic plan** | In PeopleSoft Enterprise Campus Solutions, an area of study—such as a major, minor, or specialization—that exists within an academic program or academic career. |
| **academic program** | In PeopleSoft Enterprise Campus Solutions, the entity to which a student applies and is admitted and from which the student graduates. |
| **accounting class** | In PeopleSoft Enterprise Performance Management, the accounting class defines how a resource is treated for generally accepted accounting practices. The Inventory class indicates whether a resource becomes part of a balance sheet account, such as inventory or fixed assets, while the Non-inventory class indicates that the resource is treated as an expense of the period during which it occurs. |
| **accounting date** | The accounting date indicates when a transaction is recognized, as opposed to the date the transaction actually occurred. The accounting date and transaction date can be the same. The accounting date determines the period in the general ledger to which the transaction is to be posted. You can only select an accounting date that falls within an open period in the ledger to which you are posting. The accounting date for an item is normally the invoice date. |
| **accounting split** | The accounting split method indicates how expenses are allocated or divided among one or more sets of accounting ChartFields. |
| **accumulator** | You use an accumulator to store cumulative values of defined items as they are processed. You can accumulate a single value over time or multiple values over time. For example, an accumulator could consist of all voluntary deductions, or all company deductions, enabling you to accumulate amounts. It allows total flexibility for time periods and values accumulated. |
| **action reason** | The reason an employee's job or employment information is updated. The action reason is entered in two parts: a personnel action, such as a promotion, termination, or change from one pay group to another—and a reason for that action. Action reasons are used by PeopleSoft Enterprise Human Resources, PeopleSoft Enterprise Benefits |

Copyright © 1992-2006, Oracle. All rights reserved.

Administration, PeopleSoft Enterprise Stock Administration, and the COBRA Administration feature of the Base Benefits business process.

**action template** In PeopleSoft Enterprise Receivables, outlines a set of escalating actions that the system or user performs based on the period of time that a customer or item has been in an action plan for a specific condition.

**activity** In PeopleSoft Enterprise Learning Management, an instance of a catalog item (sometimes called a class) that is available for enrollment. The activity defines such things as the costs that are associated with the offering, enrollment limits and deadlines, and waitlisting capacities.

In PeopleSoft Enterprise Performance Management, the work of an organization and the aggregation of actions that are used for activity-based costing.

In PeopleSoft Enterprise Project Costing, the unit of work that provides a further breakdown of projects—usually into specific tasks.

In PeopleSoft Workflow, a specific transaction that you might need to perform in a business process. Because it consists of the steps that are used to perform a transaction, it is also known as a step map.

**address usage** In PeopleSoft Enterprise Campus Solutions, a grouping of address types defining the order in which the address types are used. For example, you might define an address usage code to process addresses in the following order: billing address, dormitory address, home address, and then work address.

**adjustment calendar** In PeopleSoft Enterprise Campus Solutions, the adjustment calendar controls how a particular charge is adjusted on a student's account when the student drops classes or withdraws from a term. The charge adjustment is based on how much time has elapsed from a predetermined date, and it is determined as a percentage of the original charge amount.

**administrative function** In PeopleSoft Enterprise Campus Solutions, a particular functional area that processes checklists, communication, and comments. The administrative function identifies which variable data is added to a person's checklist or communication record when a specific checklist code, communication category, or comment is assigned to the student. This key data enables you to trace that checklist, communication, or comment back to a specific processing event in a functional area.

**admit type** In PeopleSoft Enterprise Campus Solutions, a designation used to distinguish first-year applications from transfer applications.

**agreement** In PeopleSoft Enterprise eSettlements, provides a way to group and specify processing options, such as payment terms, pay from a bank, and notifications by a buyer and supplier location combination.

**allocation rule** In PeopleSoft Enterprise Incentive Management, an expression within compensation plans that enables the system to assign transactions to nodes and participants. During transaction allocation, the allocation engine traverses the compensation structure from the current node to the root node, checking each node for plans that contain allocation rules.

**alternate account** A feature in PeopleSoft Enterprise General Ledger that enables you to create a statutory chart of accounts and enter statutory account transactions at the detail transaction level, as required for recording and reporting by some national governments.

**analysis database** In PeopleSoft Enterprise Campus Solutions, database tables that store large amounts of student information that may not appear in standard report formats. The analysis database tables contain keys for all objects in a report that an application program can use to reference other student-record objects that are not contained in the printed report. For instance, the analysis database contains data on courses that are considered


Copyright © 1992-2006, Oracle. All rights reserved.

| | |
|---|---|
| | for satisfying a requirement but that are rejected. It also contains information on courses captured by global limits. An analysis database is used in PeopleSoft Enterprise Academic Advisement. |
| **Application Messaging** | PeopleSoft Application Messaging enables applications within the PeopleSoft Enterprise product family to communicate synchronously or asynchronously with other PeopleSoft Enterprise and third-party applications. An application message defines the records and fields to be published or subscribed to. |
| **AR specialist** | Abbreviation for *receivables specialist.* In PeopleSoft Enterprise Receivables, an individual in who tracks and resolves deductions and disputed items. |
| **arbitration plan** | The arbiter when multiple price rules match the transaction. This plan determines the order in which the price rules are applied to the transaction base price. |
| **assessment rule** | In PeopleSoft Enterprise Receivables, a user-defined rule that the system uses to evaluate the condition of a customer's account or of individual items to determine whether to generate a follow-up action. |
| **asset class** | An asset group used for reporting purposes. It can be used in conjunction with the asset category to refine asset classification. |
| **attribute/value pair** | In PeopleSoft Enterprise Directory Interface, relates the data that makes up an entry in the directory information tree. |
| **auction event** | In PeopleSoft Strategic Sourcing, a sourcing event where bidders actively compete against one another to achieve the best price or score. |
| **audience** | In PeopleSoft Enterprise Campus Solutions, a segment of the database that relates to an initiative, or a membership organization that is based on constituent attributes rather than a dues-paying structure. Examples of audiences include the Class of '65 and Undergraduate Arts & Sciences. |
| **authentication server** | A server that is set up to verify users of the system. |
| **base time period** | In PeopleSoft Enterprise Business Planning, the lowest level time period in a calendar. |
| **benchmark job** | In PeopleSoft Enterprise Workforce Analytics Solution, a benchmark job is a job code for which there is corresponding salary survey data from published, third-party sources. |
| **bid response** | In PeopleSoft Strategic Sourcing, the response by a bidder to an event. |
| **billing career** | In PeopleSoft Enterprise Campus Solutions, the one career under which other careers are grouped for billing purposes if a student is active simultaneously in multiple careers. |
| **bio bit or bio brief** | In PeopleSoft Enterprise Campus Solutions, a report that summarizes information stored in the system about a particular constituent. You can generate standard or specialized reports. |
| **book** | In PeopleSoft Enterprise Asset Management, used for storing financial and tax information, such as costs, depreciation attributes, and retirement information on assets. |
| **branch** | A tree node that rolls up to nodes above it in the hierarchy, as defined in PeopleSoft Tree Manager. |
| **budgetary account only** | An account used by the system only and not by users; this type of account does not accept transactions. You can only budget with this account. Formerly called "system-maintained account." |

Copyright © 1992-2006, Oracle. All rights reserved.

**budget check** In commitment control, the processing of source transactions against control budget ledgers, to see if they pass, fail, or pass with a warning.

**budget control** In commitment control, budget control ensures that commitments and expenditures don't exceed budgets. It enables you to track transactions against corresponding budgets and terminate a document's cycle if the defined budget conditions are not met. For example, you can prevent a purchase order from being dispatched to a vendor if there are insufficient funds in the related budget to support it.

**budget period** The interval of time (such as 12 months or 4 quarters) into which a period is divided for budgetary and reporting purposes. The ChartField allows maximum flexibility to define operational accounting time periods without restriction to only one calendar.

**business activity** The name of a subset of a detailed business process. This might be a specific transaction, task, or action that you perform in a business process.

**business event** In PeopleSoft Enterprise Receivables, defines the processing characteristics for the Receivable Update process for a draft activity.

In PeopleSoft Enterprise Sales Incentive Management, an original business transaction or activity that may justify the creation of a PeopleSoft Enterprise Incentive Management event (a sale, for example).

**business process** A standard set of 17 business processes are defined and maintained by the PeopleSoft Enterprise product families and are supported by the Business Process Engineering group. An example of a business process is Order Fulfillment, which is a business process that manages sales orders and contracts, inventory, billing, and so forth.

See also *detailed business process.*

**business unit constraints** In PeopleSoft Strategic Sourcing, these constraints apply to a selected Strategic Sourcing business unit. Spend is tracked across all of the events within the selected Strategic Sourcing business unit.

**business task** The name of the specific function depicted in one of the business processes.

**business unit** A corporation or a subset of a corporation that is independent with regard to one or more operational or accounting functions.

**buyer** In PeopleSoft Enterprise eSettlements, an organization (or business unit, as opposed to an individual) that transacts with suppliers (vendors) within the system. A buyer creates payments for purchases that are made in the system.

**buy event** In PeopleSoft Strategic Sourcing, for event creators, the purchase of goods or services, most typically associated with a request for quote, proposal, or reverse auction.For bidders, the sale of goods or services.

**campus** In PeopleSoft Enterprise Campus Solutions, an entity that is usually associated with a distinct physical administrative unit, that belongs to a single academic institution, that uses a unique course catalog, and that produces a common transcript for students within the same academic career.

**cash drawer** A repository for monies and payments taken locally.

**catalog item** In PeopleSoft Enterprise Learning Management, a specific topic that a learner can study and have tracked. For example, "Introduction to Microsoft Word." A catalog item contains general information about the topic and includes a course code, description, categorization, keywords, and delivery methods. A catalog item can have one or more learning activities.

**catalog map** In PeopleSoft Enterprise Catalog Management, translates values from the catalog source data to the format of the company's catalog.


Copyright © 1992-2006, Oracle. All rights reserved.

**catalog partner** In PeopleSoft Enterprise Catalog Management, shares responsibility with the enterprise catalog manager for maintaining catalog content.

**categorization** Associates partner offerings with catalog offerings and groups them into enterprise catalog categories.

**category** In PeopleSoft Enterprise Campus Solutions, a broad grouping to which specific comments or communications (contexts) are assigned. Category codes are also linked to 3C access groups so that you can assign data-entry or view-only privileges across functions.

**channel** In PeopleSoft MultiChannel Framework, email, chat, voice (computer telephone integration [CTI]), or a generic event.

**ChartField** A field that stores a chart of accounts, resources, and so on, depending on the PeopleSoft Enterprise application. ChartField values represent individual account numbers, department codes, and so forth.

**ChartField balancing** You can require specific ChartFields to match up (balance) on the debit and the credit side of a transaction.

**ChartField combination edit** The process of editing journal lines for valid ChartField combinations based on user-defined rules.

**ChartKey** One or more fields that uniquely identify each row in a table. Some tables contain only one field as the key, while others require a combination.

**checkbook** In PeopleSoft Enterprise Promotions Management, enables you to view financial data (such as planned, incurred, and actual amounts) that is related to funds and trade promotions.

**checklist code** In PeopleSoft Enterprise Campus Solutions, a code that represents a list of planned or completed action items that can be assigned to a staff member, volunteer, or unit. Checklists enable you to view all action assignments on one page.

**claimback** In the wholesale distribution industry, a contract between supplier and distributor, in which monies are paid to the distributor on the sale of specified products or product groups to targeted customers or customer groups.

**class** In PeopleSoft Enterprise Campus Solutions, a specific offering of a course component within an academic term.

See also *course.*

**Class ChartField** A ChartField value that identifies a unique appropriation budget key when you combine it with a fund, department ID, and program code, as well as a budget period. Formerly called *sub-classification.*

**clearance** In PeopleSoft Enterprise Campus Solutions, the period of time during which a constituent in PeopleSoft Enterprise Contributor Relations is approved for involvement in an initiative or an action. Clearances are used to prevent development officers from making multiple requests to a constituent during the same time period.

**clone** In PeopleCode, to make a unique copy. In contrast, to *copy* may mean making a new reference to an object, so if the underlying object is changed, both the copy and the original change.

**cohort** In PeopleSoft Enterprise Campus Solutions, the highest level of the three-level classification structure that you define for enrollment management. You can define a cohort level, link it to other levels, and set enrollment target numbers for it.

See also *population* and *division*.

Copyright © 1992-2006, Oracle. All rights reserved.

**collection** To make a set of documents available for searching in Verity, you must first create at least one collection. A collection is set of directories and files that allow search application users to use the Verity search engine to quickly find and display source documents that match search criteria. A collection is a set of statistics and pointers to the source documents, stored in a proprietary format on a file server. Because a collection can only store information for a single location, PeopleTools maintains a set of collections (one per language code) for each search index object.

**collection rule** In PeopleSoft Enterprise Receivables, a user-defined rule that defines actions to take for a customer based on both the amount and the number of days past due for outstanding balances.

**comm key** See *communication key.*

**communication key** In PeopleSoft Enterprise Campus Solutions, a single code for entering a combination of communication category, communication context, communication method, communication direction, and standard letter code. Communication keys (also called *comm keys* or *speed keys*) can be created for background processes as well as for specific users.

**compensation object** In PeopleSoft Enterprise Incentive Management, a node within a compensation structure. Compensation objects are the building blocks that make up a compensation structure's hierarchical representation.

**compensation structure** In PeopleSoft Enterprise Incentive Management, a hierarchical relationship of compensation objects that represents the compensation-related relationship between the objects.

**component interface** A component interface is a set of application programming interfaces (APIs) that you can use to access and modify PeopleSoft Enterprise database information using a program instead of the PeopleSoft client.

**condition** In PeopleSoft Enterprise Receivables, occurs when there is a change of status for a customer's account, such as reaching a credit limit or exceeding a user-defined balance due.

**configuration parameter catalog** Used to configure an external system with PeopleSoft Enterprise. For example, a configuration parameter catalog might set up configuration and communication parameters for an external server.

**configuration plan** In PeopleSoft Enterprise Incentive Management, configuration plans hold allocation information for common variables (not incentive rules) and are attached to a node without a participant. Configuration plans are not processed by transactions.

**constituents** In PeopleSoft Enterprise Campus Solutions, friends, alumni, organizations, foundations, or other entities affiliated with the institution, and about which the institution maintains information. The constituent types delivered with PeopleSoft Enterprise Contributor Relations Solutions are based on those defined by the Council for the Advancement and Support of Education (CASE).

**constraint** A business policy or rule that affects how a sourcing event is awarded. There are three types of constraints: business, global, and event.

**content reference** Content references are pointers to content registered in the portal registry. These are typically either URLs or iScripts. Content references fall into three categories: target content, templates, and template pagelets.

**context** In PeopleCode, determines which buffer fields can be contextually referenced and which is the current row of data on each scroll level when a PeopleCode program is running.

In PeopleSoft Enterprise Campus Solutions, a specific instance of a comment or communication. One or more contexts are assigned to a category, which you link to

 Copyright © 1992-2006, Oracle. All rights reserved.

| | |
|---|---|
| | 3C access groups so that you can assign data-entry or view-only privileges across functions.<br><br>In PeopleSoft Enterprise Incentive Management, a mechanism that is used to determine the scope of a processing run. PeopleSoft Enterprise Incentive Management uses three types of context: plan, period, and run-level. |
| **control table** | Stores information that controls the processing of an application. This type of processing might be consistent throughout an organization, or it might be used only by portions of the organization for more limited sharing of data. |
| **cost plus contract line** | A rate-based contract line associated with a fee component of Award, Fixed, Incentive, or Other. Rate-based contract lines associated with a fee type of None are not considered cost-plus contract lines. |
| **cost plus pricing** | In PeopleSoft Enterprise Pricer, a pricing method that begins with cost of goods as the basis. |
| **cost profile** | A combination of a receipt cost method, a cost flow, and a deplete cost method. A profile is associated with a cost book and determines how items in that book are valued, as well as how the material movement of the item is valued for the book. |
| **cost row** | A cost transaction and amount for a set of ChartFields. |
| **counter sale** | A face-to-face customer transaction where the customer typically selects items from the storefront or picks up products that they ordered ahead of time. Customers pay for the goods at the counter and take the goods with them instead of having the goods shipped from a warehouse. |
| **course** | In PeopleSoft Enterprise Campus Solutions, a course that is offered by a school and that is typically described in a course catalog. A course has a standard syllabus and credit level; however, these may be modified at the class level. Courses can contain multiple components such as lecture, discussion, and lab.<br><br>See also *class.* |
| **course share set** | In PeopleSoft Enterprise Campus Solutions, a tag that defines a set of requirement groups that can share courses. Course share sets are used in PeopleSoft Enterprise Academic Advisement. |
| **current learning** | In PeopleSoft Enterprise Learning Management, a self-service repository for all of a learner's in-progress learning activities and programs. |
| **data acquisition** | In PeopleSoft Enterprise Incentive Management, the process during which raw business transactions are acquired from external source systems and fed into the operational data store (ODS). |
| **data cube** | In PeopleSoft Analytic Calculation Engine, a data cube is a container for one kind of data (such as Sales data) and works with in tandem with one or more dimensions. Dimensions and data cubes in PeopleSoft Analytic Calculation Engine are unrelated to dimensions and online analytical processing (OLAP) cubes in PeopleSoft Cube Manager. |
| **data elements** | Data elements, at their simplest level, define a subset of data and the rules by which to group them.<br><br>For Workforce Analytics, data elements are rules that tell the system what measures to retrieve about your workforce groups. |
| **dataset** | A data grouping that enables role-based filtering and distribution of data. You can limit the range and quantity of data that is displayed for a user by associating dataset rules with user roles. The result of dataset rules is a set of data that is appropriate for the user's roles. |

Copyright © 1992-2006, Oracle. All rights reserved.

**delivery method**

In PeopleSoft Enterprise Learning Management, identifies the primary type of delivery method in which a particular learning activity is offered. Also provides default values for the learning activity, such as cost and language. This is primarily used to help learners search the catalog for the type of delivery from which they learn best. Because PeopleSoft Enterprise Learning Management is a blended learning system, it does not enforce the delivery method.

In PeopleSoft Enterprise Supply Chain Management, identifies the method by which goods are shipped to their destinations (such as truck, air, and rail). The delivery method is specified when creating shipment schedules.

**delivery method type**

In PeopleSoft Enterprise Learning Management, identifies how learning activities can be delivered—for example, through online learning, classroom instruction, seminars, books, and so forth—in an organization. The type determines whether the delivery method includes scheduled components.

**detailed business process**

A subset of the business process. For example, the detailed business process named Determine Cash Position is a subset of the business process called Cash Management.

**dimension**

In PeopleSoft Analytic Calculation Engine, a dimension contains a list of one kind of data that can span various contexts, and it is a basic component of an analytic model. Within the analytic model, a dimension is attached to one or more data cubes. In PeopleSoft Cube Manager, a dimension is the most basic component of an OLAP cube and specifies the PeopleSoft metadata to be used to create the dimension's rollup structure. Dimensions and data cubes in PeopleSoft Analytic Calculation Engine are unrelated to dimensions and OLAP cubes in PeopleSoft Cube Manager.

**direct receipt**

Items shipped from a warehouse or vendor to another warehouse.

**direct ship**

Items shipped from the vendor or warehouse directly to the customer (formerly referred to as *drop ship*).

**directory information tree**

In PeopleSoft Enterprise Directory Interface, the representation of a directory's hierarchical structure.

**division**

In PeopleSoft Enterprise Campus Solutions, the lowest level of the three-level classification structure that you define in PeopleSoft Enterprise Recruiting and Admissions for enrollment management. You can define a division level, link it to other levels, and set enrollment target numbers for it.

See also *population* and *cohort*.

**document sequencing**

A flexible method that sequentially numbers the financial transactions (for example, bills, purchase orders, invoices, and payments) in the system for statutory reporting and for tracking commercial transaction activity.

**dynamic detail tree**

A tree that takes its detail values—dynamic details—directly from a table in the database, rather than from a range of values that are entered by the user.

**edit table**

A table in the database that has its own record definition, such as the Department table. As fields are entered into a PeopleSoft Enterprise application, they can be validated against an edit table to ensure data integrity throughout the system.

**effective date**

A method of dating information in PeopleSoft Enterprise applications. You can predate information to add historical data to your system, or postdate information in order to enter it before it actually goes into effect. By using effective dates, you don't delete values; you enter a new value with a current effective date.

**EIM ledger**

Abbreviation for *Enterprise Incentive Management ledger.* In PeopleSoft Enterprise Incentive Management, an object to handle incremental result gathering within the scope of a participant. The ledger captures a result set with all of the appropriate traces to the data origin and to the processing steps of which it is a result.


Copyright © 1992-2006, Oracle. All rights reserved.

**elimination set** In PeopleSoft Enterprise General Ledger, a related group of intercompany accounts that is processed during consolidations.

**entry event** In PeopleSoft Enterprise General Ledger, Receivables, Payables, Purchasing, and Billing, a business process that generates multiple debits and credits resulting from single transactions to produce standard, supplemental accounting entries.

**equitization** In PeopleSoft Enterprise General Ledger, a business process that enables parent companies to calculate the net income of subsidiaries on a monthly basis and adjust that amount to increase the investment amount and equity income amount before performing consolidations.

**equity item limit** In PeopleSoft Enterprise Campus Solutions, the amounts of funds set by the institution to be awarded with discretionary or gift funds. The limit could be reduced by amounts equal to such things as expected family contribution (EFC) or parent contribution. Students are packaged by Equity Item Type Groups and Related Equity Item Types. This limit can be used to assure that similar student populations are packaged equally.

**event** A predefined point either in the Component Processor flow or in the program flow. As each point is encountered, the event activates each component, triggering any PeopleCode program that is associated with that component and that event. Examples of events are FieldChange, SavePreChange, and RowDelete.

In PeopleSoft Enterprise Human Resources, also refers to an incident that affects benefits eligibility.

**event constraints** In PeopleSoft Strategic Sourcing, these constraints are associated with a specific sourcing event. Spend is tracked within the selected event.

**event propagation process** In PeopleSoft Enterprise Sales Incentive Management, a process that determines, through logic, the propagation of an original PeopleSoft Enterprise Incentive Management event and creates a derivative (duplicate) of the original event to be processed by other objects. PeopleSoft Enterprise Enterprise Sales Incentive Management uses this mechanism to implement splits, roll-ups, and so on. Event propagation determines who receives the credit.

**exception** In PeopleSoft Enterprise Receivables, an item that either is a deduction or is in dispute.

**exclusive pricing** In PeopleSoft Enterprise Order Management, a type of arbitration plan that is associated with a price rule. Exclusive pricing is used to price sales order transactions.

**fact** In PeopleSoft Enterprise applications, facts are numeric data values from fields from a source database as well as an analytic application. A fact can be anything you want to measure your business by, for example, revenue, actual, budget data, or sales numbers. A fact is stored on a fact table.

**financial aid term** In PeopleSoft Enterprise Campus Solutions, a combination of a period of time that the school determines as an instructional accounting period and an academic career. It is created and defined during the setup process. Only terms eligible for financial aid are set up for each financial aid career.

**financial sanctions** For U.S. based companies and their foreign subsidiaries, a federal regulation from the Office of Foreign Assets Control (OFAC) requires that vendors be validated against a Specially Designated Nationals (SDN) list prior to payment.

For PeopleSoft Payables, eSettlements, Cash Management, and Order to Cash, you can validate your vendors against any financial sanctions list (for example, the SDN list, a European Union list, and so on).

**forecast item** A logical entity with a unique set of descriptive demand and forecast data that is used as the basis to forecast demand. You create forecast items for a wide range of uses, but they ultimately represent things that you buy, sell, or use in your organization and for which you require a predictable usage.

Copyright © 1992-2006, Oracle. All rights reserved.

**fund** In PeopleSoft Enterprise Promotions Management, a budget that can be used to fund promotional activity. There are four funding methods: top down, fixed accrual, rolling accrual, and zero-based accrual.

**gap** In PeopleSoft Enterprise Campus Solutions, an artificial figure that sets aside an amount of unmet financial aid need that is not funded with Title IV funds. A gap can be used to prevent fully funding any student to conserve funds, or it can be used to preserve unmet financial aid need so that institutional funds can be awarded.

**generic process type** In PeopleSoft Process Scheduler, process types are identified by a generic process type. For example, the generic process type SQR includes all SQR process types, such as SQR process and SQR report.

**gift table** In PeopleSoft Enterprise Campus Solutions, a table or so-called *donor pyramid* describing the number and size of gifts that you expect will be needed to successfully complete the campaign in PeopleSoft Enterprise Contributor Relations. The gift table enables you to estimate the number of donors and prospects that you need at each gift level to reach the campaign goal.

**GDS** Abbreviation for *Global Distribution System.* Broad-based term to describe all computer reservation systems for making travel plans.

**GL business unit** Abbreviation for *general ledger business unit.* A unit in an organization that is an independent entity for accounting purposes. It maintains its own set of accounting books.

See also *business unit.*

**GL entry template** Abbreviation for *general ledger entry template.* In PeopleSoft Enterprise Campus Solutions, a template that defines how a particular item is sent to the general ledger. An item-type maps to the general ledger, and the GL entry template can involve multiple general ledger accounts. The entry to the general ledger is further controlled by high-level flags that control the summarization and the type of accounting—that is, accrual or cash.

**GL Interface process** Abbreviation for *General Ledger Interface process.* In PeopleSoft Enterprise Campus Solutions, a process that is used to send transactions from PeopleSoft Enterprise Student Financials to the general ledger. Item types are mapped to specific general ledger accounts, enabling transactions to move to the general ledger when the GL Interface process is run.

**global constraints** In PeopleSoft Strategic Sourcing, these constraints apply across multiple Strategic Sourcing business units. Spend is tracked across all of the events from the multiple Strategic Sourcing business units.

**group** In PeopleSoft Enterprise Billing and Receivables, a posting entity that comprises one or more transactions (items, deposits, payments, transfers, matches, or write-offs).

In PeopleSoft Enterprise Human Resources Management and Supply Chain Management, any set of records that are associated under a single name or variable to run calculations in PeopleSoft business processes. In PeopleSoft Enterprise Time and Labor, for example, employees are placed in groups for time reporting purposes.

**ideal response** In PeopleSoft Strategic Sourcing, a question that requires the response to match the ideal value for the bid to be considered eligible for award. If the response does not match the ideal value, you can still submit the bid, but it will be disqualified and ineligible for award.

**incentive object** In PeopleSoft Enterprise Incentive Management, the incentive-related objects that define and support the PeopleSoft Enterprise Incentive Management calculation process and results, such as plan templates, plans, results data, and user interaction objects.


Copyright © 1992-2006, Oracle. All rights reserved.

**incentive rule** In PeopleSoft Enterprise Sales Incentive Management, the commands that act on transactions and turn them into compensation. A rule is one part in the process of turning a transaction into compensation.

**incur** In PeopleSoft Enterprise Promotions Management, to become liable for a promotional payment. In other words, you owe that amount to a customer for promotional activities.

**initiative** In PeopleSoft Enterprise Campus Solutions, the basis from which all advancement plans are executed. It is an organized effort targeting a specific constituency, and it can occur over a specified period of time with specific purposes and goals. An initiative can be a campaign, an event, an organized volunteer effort, a membership drive, or any other type of effort defined by the institution. Initiatives can be multipart, and they can be related to other initiatives. This enables you to track individual parts of an initiative, as well as entire initiatives.

**inquiry access** In PeopleSoft Enterprise Campus Solutions, a type of security access that permits the user only to view data.

See also *update access.*

**institution** In PeopleSoft Enterprise Campus Solutions, an entity (such as a university or college) that is independent of other similar entities and that has its own set of rules and business processes.

**integration** A relationship between two compatible integration points that enables communication to take place between systems. Integrations enable PeopleSoft Enterprise applications to work seamlessly with other PeopleSoft Enterprise applications or with third-party systems or software.

**integration point** An interface that a system uses to communicate with another PeopleSoft Enterprise application or an external application.

**integration set** A logical grouping of integrations that applications use for the same business purpose. For example, the integration set ADVANCED_SHIPPING_ORDER contains all of the integrations that notify a customer that an order has shipped.

**item** In PeopleSoft Enterprise Inventory, a tangible commodity that is stored in a business unit (shipped from a warehouse).

In PeopleSoft Enterprise Demand Planning, Inventory Policy Planning, and Supply Planning, a noninventory item that is designated as being used for planning purposes only. It can represent a family or group of inventory items. It can have a planning bill of material (BOM) or planning routing, and it can exist as a component on a planning BOM. A planning item cannot be specified on a production or engineering BOM or routing, and it cannot be used as a component in a production. The quantity on hand will never be maintained.

In PeopleSoft Enterprise Receivables, an individual receivable. An item can be an invoice, a credit memo, a debit memo, a write-off, or an adjustment.

**item shuffle** In PeopleSoft Enterprise Campus Solutions, a process that enables you to change a payment allocation without having to reverse the payment.

**itinerary** In PeopleSoft Expenses, a collection of travel reservations. Itineraries can have reservations that are selected and reserved with the travel vendor. These itineraries are not yet paid for and can be referred to as *pending reservations.* Reservations that have been paid for are referred to as *confirmed reservations.*

**joint communication** In PeopleSoft Enterprise Campus Solutions, one letter that is addressed jointly to two people. For example, a letter might be addressed to both Mr. Sudhir Awat and Ms. Samantha Mortelli. A relationship must be established between the two individuals in the database, and at least one of the individuals must have an ID in the database.

Copyright © 1992-2006, Oracle. All rights reserved.

**keyword** In PeopleSoft Enterprise Campus Solutions, a term that you link to particular elements within PeopleSoft Enterprise Student Financials, Financial Aid, and Contributor Relations. You can use keywords as search criteria that enable you to locate specific records in a search dialog box.

**KPI** An abbreviation for *key performance indicator.* A high-level measurement of how well an organization is doing in achieving critical success factors. This defines the data value or calculation upon which an assessment is determined.

**KVI** Abbreviation for *Known Value Item.* Term used for products or groups of products where the selling price cannot be reduced or increased.

**landlord** In PeopleSoft Real Estate Management, an entity that owns real estate and leases the real estate to tenants.

**LDIF file** Abbreviation for *Lightweight Directory Access Protocol (LDAP) Data Interchange Format file.* Contains discrepancies between PeopleSoft Enterprise data and directory data.

**learner group** In PeopleSoft Enterprise Learning Management, a group of learners who are linked to the same learning environment. Members of the learner group can share the same attributes, such as the same department or job code. Learner groups are used to control access to and enrollment in learning activities and programs. They are also used to perform group enrollments and mass enrollments in the back office.

**learning components** In PeopleSoft Enterprise Learning Management, the foundational building blocks of learning activities. PeopleSoft Enterprise Learning Management supports six basic types of learning components: web-based, session, webcast, test, survey, and assignment. One or more of these learning component types compose a single learning activity.

**learning environment** In PeopleSoft Enterprise Learning Management, identifies a set of categories and catalog items that can be made available to learner groups. Also defines the default values that are assigned to the learning activities and programs that are created within a particular learning environment. Learning environments provide a way to partition the catalog so that learners see only those items that are relevant to them.

**learning history** In PeopleSoft Enterprise Learning Management, a self-service repository for all of a learner's completed learning activities and programs.

**lease** In PeopleSoft Real Estate Management, a legally binding agreement between a landlord and a tenant, where the tenant rents all or part of a physical property from the landlord.

**lease abstract** In PeopleSoft Real Estate Management, a summarized version of the complete lease contract with only the important terms. The lease abstract usually fits on one page and does not include legal terminology.

**ledger mapping** You use ledger mapping to relate expense data from general ledger accounts to resource objects. Multiple ledger line items can be mapped to one or more resource IDs. You can also use ledger mapping to map dollar amounts (referred to as *rates*) to business units. You can map the amounts in two different ways: an actual amount that represents actual costs of the accounting period, or a budgeted amount that can be used to calculate the capacity rates as well as budgeted model results. In PeopleSoft Enterprise Warehouse, you can map general ledger accounts to the EW Ledger table.

**library section** In PeopleSoft Enterprise Incentive Management, a section that is defined in a plan (or template) and that is available for other plans to share. Changes to a library section are reflected in all plans that use it.

**line** In PeopleSoft Strategic Sourcing, an individual item or service upon which there can be a bid.


Copyright © 1992-2006, Oracle. All rights reserved.

**linked section** In PeopleSoft Enterprise Incentive Management, a section that is defined in a plan template but appears in a plan. Changes to linked sections propagate to plans using that section.

**linked variable** In PeopleSoft Enterprise Incentive Management, a variable that is defined and maintained in a plan template and that also appears in a plan. Changes to linked variables propagate to plans using that variable.

**LMS** Abbreviation for *learning management system.* In PeopleSoft Enterprise Campus Solutions, LMS is a PeopleSoft Enterprise Student Records feature that provides a common set of interoperability standards that enable the sharing of instructional content and data between learning and administrative environments.

**load** In PeopleSoft Enterprise Inventory, identifies a group of goods that are shipped together. Load management is a feature of PeopleSoft Enterprise Inventory that is used to track the weight, the volume, and the destination of a shipment.

**local functionality** In PeopleSoft Enterprise HRMS, the set of information that is available for a specific country. You can access this information when you click the appropriate country flag in the global window, or when you access it by a local country menu.

**location** Locations enable you to indicate the different types of addresses—for a company, for example, one address to receive bills, another for shipping, a third for postal deliveries, and a separate street address. Each address has a different location number. The primary location—indicated by a *1*—is the address you use most often and may be different from the main address.

**logistical task** In PeopleSoft Enterprise Services Procurement, an administrative task that is related to hiring a service provider. Logistical tasks are linked to the service type on the work order so that different types of services can have different logistical tasks. Logistical tasks include both preapproval tasks (such as assigning a new badge or ordering a new laptop) and postapproval tasks (such as scheduling orientation or setting up the service provider email). The logistical tasks can be mandatory or optional. Mandatory preapproval tasks must be completed before the work order is approved. Mandatory postapproval tasks, on the other hand, must be completed before a work order is released to a service provider.

**market template** In PeopleSoft Enterprise Incentive Management, additional functionality that is specific to a given market or industry and is built on top of a product category.

**mass change** In PeopleSoft Enterprise Campus Solutions, mass change is a SQL generator that can be used to create specialized functionality. Using mass change, you can set up a series of Insert, Update, or Delete SQL statements to perform business functions that are specific to the institution.

See also *3C engine.*

**match group** In PeopleSoft Enterprise Receivables, a group of receivables items and matching offset items. The system creates match groups by using user-defined matching criteria for selected field values.

**MCF server** Abbreviation for *PeopleSoft MultiChannel Framework server.* Comprises the universal queue server and the MCF log server. Both processes are started when *MCF Servers* is selected in an application server domain configuration.

**merchandising activity** In PeopleSoft Enterprise Promotions Management, a specific discount type that is associated with a trade promotion (such as off-invoice, billback or rebate, or lump-sum payment) that defines the performance that is required to receive the discount. In the industry, you may know this as an offer, a discount, a merchandising event, an event, or a tactic.

Copyright © 1992-2006, Oracle. All rights reserved.

**meta-SQL** Meta-SQL constructs expand into platform-specific SQL substrings. They are used in functions that pass SQL strings, such as in SQL objects, the SQLExec function, and PeopleSoft Application Engine programs.

**metastring** Metastrings are special expressions included in SQL string literals. The metastrings, prefixed with a percent (%) symbol, are included directly in the string literals. They expand at run time into an appropriate substring for the current database platform.

**multibook** In PeopleSoft Enterprise General Ledger, multiple ledgers having multiple-base currencies that are defined for a business unit, with the option to post a single transaction to all base currencies (all ledgers) or to only one of those base currencies (ledgers).

**multicurrency** The ability to process transactions in a currency other than the business unit's base currency.

**national allowance** In PeopleSoft Enterprise Promotions Management, a promotion at the corporate level that is funded by nondiscretionary dollars. In the industry, you may know this as a national promotion, a corporate promotion, or a corporate discount.

**NDP** Abbreviation for *Non-Discountable Products.* Term used for products or groups of products where the selling price cannot be decreased.

**need** In PeopleSoft Enterprise Campus Solutions, the difference between the cost of attendance (COA) and the expected family contribution (EFC). It is the gap between the cost of attending the school and the student's resources. The financial aid package is based on the amount of financial need. The process of determining a student's need is called *need analysis.*

**node-oriented tree** A tree that is based on a detail structure, but the detail values are not used.

**Optimization Engine** A PeopleTools component that Strategic Sourcing leverages to evaluate bids and determine an ideal award allocation. The award recommendation is based on maximizing the value while adhering to purchasing and company objectives and constraints.

**pagelet** Each block of content on the home page is called a pagelet. These pagelets display summary information within a small rectangular area on the page. The pagelet provide users with a snapshot of their most relevant PeopleSoft Enterprise and non-PeopleSoft Enterprise content.

**participant** In PeopleSoft Enterprise Incentive Management, participants are recipients of the incentive compensation calculation process.

**participant object** Each participant object may be related to one or more compensation objects.

See also *compensation object.*

**partner** A company that supplies products or services that are resold or purchased by the enterprise.

**pay cycle** In PeopleSoft Enterprise Payables, a set of rules that define the criteria by which it should select scheduled payments for payment creation.

**payment shuffle** In PeopleSoft Enterprise Campus Solutions, a process allowing payments that have been previously posted to a student's account to be automatically reapplied when a higher priority payment is posted or the payment allocation definition is changed.

**pending item** In PeopleSoft Enterprise Receivables, an individual receivable (such as an invoice, a credit memo, or a write-off) that has been entered in or created by the system, but hasn't been posted.


Copyright © 1992-2006, Oracle. All rights reserved.

**PeopleCode** PeopleCode is a proprietary language, executed by the PeopleSoft Enterprise component processor. PeopleCode generates results based on existing data or user actions. By using various tools provided with PeopleTools, external services are available to all PeopleSoft Enterprise applications wherever PeopleCode can be executed.

**PeopleCode event** See *event.*

**PeopleSoft Pure Internet Architecture** The fundamental architecture on which PeopleSoft 8 applications are constructed, consisting of a relational database management system (RDBMS), an application server, a web server, and a browser.

**performance measurement** In PeopleSoft Enterprise Incentive Management, a variable used to store data (similar to an aggregator, but without a predefined formula) within the scope of an incentive plan. Performance measures are associated with a plan calendar, territory, and participant. Performance measurements are used for quota calculation and reporting.

**period context** In PeopleSoft Enterprise Incentive Management, because a participant typically uses the same compensation plan for multiple periods, the period context associates a plan context with a specific calendar period and fiscal year. The period context references the associated plan context, thus forming a chain. Each plan context has a corresponding set of period contexts.

**person of interest** A person about whom the organization maintains information but who is not part of the workforce.

**personal portfolio** In PeopleSoft Enterprise Campus Solutions, the user-accessible menu item that contains an individual's name, address, telephone number, and other personal information.

**phase** A level 1 task, meaning that if a task had subtasks, the level 1 task would be considered the phase.

**pickup quantity** The product quantity that the customer is taking with them from the counter sales environment.

**plan** In PeopleSoft Enterprise Sales Incentive Management, a collection of allocation rules, variables, steps, sections, and incentive rules that instruct the PeopleSoft Enterprise Incentive Management engine in how to process transactions.

**plan context** In PeopleSoft Enterprise Incentive Management, correlates a participant with the compensation plan and node to which the participant is assigned, enabling the PeopleSoft Enterprise Incentive Management system to find anything that is associated with the node and that is required to perform compensation processing. Each participant, node, and plan combination represents a unique plan context—if three participants are on a compensation structure, each has a different plan context. Configuration plans are identified by plan contexts and are associated with the participants that refer to them.

**plan template** In PeopleSoft Enterprise Incentive Management, the base from which a plan is created. A plan template contains common sections and variables that are inherited by all plans that are created from the template. A template may contain steps and sections that are not visible in the plan definition.

**planned learning** In PeopleSoft Enterprise Learning Management, a self-service repository for all of a learner's planned learning activities and programs.

**planning instance** In PeopleSoft Enterprise Supply Planning, a set of data (business units, items, supplies, and demands) constituting the inputs and outputs of a supply plan.

**population** In PeopleSoft Enterprise Campus Solutions, the middle level of the three-level classification structure that you define in PeopleSoft Enterprise Recruiting and

Copyright © 1992-2006, Oracle. All rights reserved.

| | |
|---|---|
| | Admissions for enrollment management. You can define a population level, link it to other levels, and set enrollment target numbers for it.<br><br>See also *division* and *cohort.* |
| **portal registry** | In PeopleSoft Enterprise applications, the portal registry is a tree-like structure in which content references are organized, classified, and registered. It is a central repository that defines both the structure and content of a portal through a hierarchical, tree-like structure of folders useful for organizing and securing content references. |
| **predecessor task** | A task that you must complete before you start another task. |
| **price breaks** | In PeopleSoft Strategic Sourcing, a price discount or surcharge that a bidder may apply based on the quantity awarded. |
| **price components** | In PeopleSoft Strategic Sourcing, the various components, such as material costs, labor costs, shipping costs, and so on that make up the overall bid price. |
| **price list** | Enables you to select products and conditions for which the price list applies to a transaction. During a transaction, the system either determines the product price based on the predefined search hierarchy for the transaction or uses the product's lowest price on any associated, active price lists. This price is used as the basis for any further discounts and surcharges. |
| **price rule** | The conditions that must be met for adjustments to be applied to the base price. Multiple rules can apply when conditions of each rule are met. |
| **price rule conditions** | Conditions that select the price-by fields, the values for the price-by fields, and the operator that determines how the price-by fields relate to the transaction. |
| **price rule key** | The fields that are available to define price rule conditions (which are used to match a transaction) on the price rule. |
| **primacy number** | In PeopleSoft Enterprise Campus Solutions, a number that the system uses to prioritize financial aid applications when students are enrolled in multiple academic careers and academic programs at the same time. The Consolidate Academic Statistics process uses the primacy number indicated for both the career and program at the institutional level to determine a student's primary career and program. The system also uses the number to determine the primary student attribute value that is used when you extract data to report on cohorts. The lowest number takes precedence. |
| **primary name type** | In PeopleSoft Enterprise Campus Solutions, the name type that is used to link the name stored at the highest level within the system to the lower-level set of names that an individual provides. |
| **process category** | In PeopleSoft Process Scheduler, processes that are grouped for server load balancing and prioritization. |
| **process group** | In PeopleSoft Enterprise Financials, a group of application processes (performed in a defined order) that users can initiate in real time, directly from a transaction entry page. |
| **process definition** | Process definitions define each run request. |
| **process instance** | A unique number that identifies each process request. This value is automatically incremented and assigned to each requested process when the process is submitted to run. |
| **process job** | You can link process definitions into a job request and process each request serially or in parallel. You can also initiate subsequent processes based on the return code from each prior request. |


Copyright © 1992-2006, Oracle. All rights reserved.

**process request** A single run request, such as a Structured Query Report (SQR), a COBOL or Application Engine program, or a Crystal report that you run through PeopleSoft Process Scheduler.

**process run control** A PeopleTools variable used to retain PeopleSoft Process Scheduler values needed at runtime for all requests that reference a run control ID. Do not confuse these with application run controls, which may be defined with the same run control ID, but only contain information specific to a given application process request.

**product** A PeopleSoft Enterprise or third-party product. PeopleSoft organizes its software products into product families and product lines. Interactive Services Repository contains information about every release of every product that PeopleSoft sells, as well as products from certified third-party companies. These products appear with the product name and release number.

**product adds** The pricing functionality where buying product A gets product B for free or at a price (formerly referred to as *giveaways*).

**product bidding** In PeopleSoft Strategic Sourcing, the placing of a bid on behalf of the bidder, up or down to the bidder's specified amount, so that the bidder can be the leading bidder.

**product category** In PeopleSoft Enterprise Incentive Management, indicates an application in the PeopleSoft Enterprise Incentive Management suite of products. Each transaction in the PeopleSoft Enterprise Incentive Management system is associated with a product category.

**product family** A group of products that are related by common functionality. The family names that can be searched using Interactive Service Repository are Oracle's PeopleSoft Enterprise, PeopleSoft EnterpriseOne, PeopleSoft World, and third-party, certified partners.

**product line** The name of a PeopleSoft Enterprise product line or the company name of a third-party certified partner. Integration Services Repository enables you to search for integration points by product line.

**programs** In PeopleSoft Enterprise Learning Management, a high-level grouping that guides the learner along a specific learning path through sections of catalog items. PeopleSoft Enterprise Learning Systems provides two types of programs—curricula and certifications.

**progress log** In PeopleSoft Enterprise Services Procurement, tracks deliverable-based projects. This is similar to the time sheet in function and process. The service provider contact uses the progress log to record and submit progress on deliverables. The progress can be logged by the activity that is performed, by the percentage of work that is completed, or by the completion of milestone activities that are defined for the project.

**project transaction** In PeopleSoft Enterprise Project Costing, an individual transaction line that represents a cost, time, budget, or other transaction row.

**promotion** In PeopleSoft Enterprise Promotions Management, a trade promotion, which is typically funded from trade dollars and used by consumer products manufacturers to increase sales volume.

**prospects** In PeopleSoft Enterprise Campus Solutions, students who are interested in applying to the institution.

In PeopleSoft Enterprise Contributor Relations, individuals and organizations that are most likely to make substantial financial commitments or other types of commitments to the institution.

**proxy bidding** In PeopleSoft Strategic Sourcing, the placing of a bid on behalf of the bidder, up or down to the bidder's specified amount, so that the bidder can be the leading bidder.

Copyright © 1992-2006, Oracle. All rights reserved.

**publishing** In PeopleSoft Enterprise Incentive Management, a stage in processing that makes incentive-related results available to participants.

**rating components** In PeopleSoft Enterprise Campus Solutions, variables used with the Equation Editor to retrieve specified populations.

**record group** A set of logically and functionally related control tables and views. Record groups help enable TableSet sharing, which eliminates redundant data entry. Record groups ensure that TableSet sharing is applied consistently across all related tables and views.

**record input VAT flag** Abbreviation for *record input value-added tax flag.* Within PeopleSoft Enterprise Purchasing, Payables, and General Ledger, this flag indicates that you are recording input VAT on the transaction. This flag, in conjunction with the record output VAT flag, is used to determine the accounting entries created for a transaction and to determine how a transaction is reported on the VAT return. For all cases within Purchasing and Payables where VAT information is tracked on a transaction, this flag is set to Yes. This flag is not used in PeopleSoft Enterprise Order Management, Billing, or Receivables, where it is assumed that you are always recording only output VAT, or in PeopleSoft Enterprise Expenses, where it is assumed that you are always recording only input VAT.

**record output VAT flag** Abbreviation for *record output value-added tax flag.*

See *record input VAT flag.*

**recname** The name of a record that is used to determine the associated field to match a value or set of values.

**recognition** In PeopleSoft Enterprise Campus Solutions, the recognition type indicates whether the PeopleSoft Enterprise Contributor Relations donor is the primary donor of a commitment or shares the credit for a donation. Primary donors receive hard credit that must total 100 percent. Donors that share the credit are given soft credit. Institutions can also define other share recognition-type values such as memo credit or vehicle credit.

**reference data** In PeopleSoft Enterprise Sales Incentive Management, system objects that represent the sales organization, such as territories, participants, products, customers, and channels.

**reference object** In PeopleSoft Enterprise Incentive Management, this dimension-type object further defines the business. Reference objects can have their own hierarchy (for example, product tree, customer tree, industry tree, and geography tree).

**reference transaction** In commitment control, a reference transaction is a source transaction that is referenced by a higher-level (and usually later) source transaction, in order to automatically reverse all or part of the referenced transaction's budget-checked amount. This avoids duplicate postings during the sequential entry of the transaction at different commitment levels. For example, the amount of an encumbrance transaction (such as a purchase order) will, when checked and recorded against a budget, cause the system to concurrently reference and relieve all or part of the amount of a corresponding pre-encumbrance transaction, such as a purchase requisition.

**regional sourcing** In PeopleSoft Enterprise Purchasing, provides the infrastructure to maintain, display, and select an appropriate vendor and vendor pricing structure that is based on a regional sourcing model where the multiple ship to locations are grouped. Sourcing may occur at a level higher than the ship to location.

**relationship object** In PeopleSoft Enterprise Incentive Management, these objects further define a compensation structure to resolve transactions by establishing associations between compensation objects and business objects.

**remote data source data** Data that is extracted from a separate database and migrated into the local database.


Copyright © 1992-2006, Oracle. All rights reserved.

**REN server** Abbreviation for *real-time event notification server* in PeopleSoft MultiChannel Framework.

**requester** In PeopleSoft Enterprise eSettlements, an individual who requests goods or services and whose ID appears on the various procurement pages that reference purchase orders.

**reservations** In PeopleSoft Expenses, travel reservations that have been placed with the travel vendor.

**reversal indicator** In PeopleSoft Enterprise Campus Solutions, an indicator that denotes when a particular payment has been reversed, usually because of insufficient funds.

**RFI event** In PeopleSoft Strategic Sourcing, a request for information.

**RFx event** In PeopleSoft Strategic Sourcing, a request for proposal or request for a quote event when bidders submit their overall best bids and during which bidders do not actively compete against one another.

**role** Describes how people fit into PeopleSoft Workflow. A role is a class of users who perform the same type of work, such as clerks or managers. Your business rules typically specify what user role needs to do an activity.

**role user** A PeopleSoft Workflow user. A person's role user ID serves much the same purpose as a user ID does in other parts of the system. PeopleSoft Workflow uses role user IDs to determine how to route worklist items to users (through an email address, for example) and to track the roles that users play in the workflow. Role users do not need PeopleSoft user IDs.

**roll up** In a tree, to roll up is to total sums based on the information hierarchy.

**run control** A run control is a type of online page that is used to begin a process, such as the batch processing of a payroll run. Run control pages generally start a program that manipulates data.

**run control ID** A unique ID to associate each user with his or her own run control table entries.

**run-level context** In PeopleSoft Enterprise Incentive Management, associates a particular run (and batch ID) with a period context and plan context. Every plan context that participates in a run has a separate run-level context. Because a run cannot span periods, only one run-level context is associated with each plan context.

**saved bid** In PeopleSoft Strategic Sourcing, a bid that has been created but not submitted. Only submitted bids are eligible for award.

**score** In PeopleSoft Strategic Sourcing, the numerical sum of answers (percentages) to bid factors on an event. Scores appear only to bidders on auction events.

**SCP SCBM XML message** Abbreviation for *Supply Chain Planning Supply Chain Business Modeler Extensible Markup Language message.* Supply Chain Business Modeler uses XML as the format for all data that it imports and exports.

**search query** You use this set of objects to pass a query string and operators to the search engine. The search index returns a set of matching results with keys to the source documents.

**search/match** In PeopleSoft Enterprise Campus Solutions and PeopleSoft Enterprise Human Resources Management Solutions, a feature that enables you to search for and identify duplicate records in the database.

**seasonal address** In PeopleSoft Enterprise Campus Solutions, an address that recurs for the same length of time at the same time of year each year until adjusted or deleted.

Copyright © 1992-2006, Oracle. All rights reserved.

**section** In PeopleSoft Enterprise Incentive Management, a collection of incentive rules that operate on transactions of a specific type. Sections enable plans to be segmented to process logical events in different sections.

**security event** In commitment control, security events trigger security authorization checking, such as budget entries, transfers, and adjustments; exception overrides and notifications; and inquiries.

**sell event** In PeopleSoft Strategic Sourcing, for event creators, the sale of goods or services most typically associated with forward auctions. For bidders, the purchase of goods or services.

**serial genealogy** In PeopleSoft Enterprise Manufacturing, the ability to track the composition of a specific, serial-controlled item.

**serial in production** In PeopleSoft Enterprise Manufacturing, enables the tracing of serial information for manufactured items. This is maintained in the Item Master record.

**service impact** In PeopleSoft Enterprise Campus Solutions, the resulting action triggered by a service indicator. For example, a service indicator that reflects nonpayment of account balances by a student might result in a service impact that prohibits registration for classes.

**service indicator** In PeopleSoft Enterprise Campus Solutions, indicates services that may be either withheld or provided to an individual. Negative service indicators indicate holds that prevent the individual from receiving specified services, such as check-cashing privileges or registration for classes. Positive service indicators designate special services that are provided to the individual, such as front-of-line service or special services for disabled students.

**session** In PeopleSoft Enterprise Campus Solutions, time elements that subdivide a term into multiple time periods during which classes are offered. In PeopleSoft Enterprise Contributor Relations, a session is the means of validating gift, pledge, membership, or adjustment data entry . It controls access to the data entered by a specific user ID. Sessions are balanced, queued, and then posted to the institution's financial system. Sessions must be posted to enter a matching gift or pledge payment, to make an adjustment, or to process giving clubs or acknowledgements.

In PeopleSoft Enterprise Learning Management, a single meeting day of an activity (that is, the period of time between start and finish times within a day). The session stores the specific date, location, meeting time, and instructor. Sessions are used for scheduled training.

**session template** In PeopleSoft Enterprise Learning Management, enables you to set up common activity characteristics that may be reused while scheduling a PeopleSoft Enterprise Learning Management activity—characteristics such as days of the week, start and end times, facility and room assignments, instructors, and equipment. A session pattern template can be attached to an activity that is being scheduled. Attaching a template to an activity causes all of the default template information to populate the activity session pattern.

**setup relationship** In PeopleSoft Enterprise Incentive Management, a relationship object type that associates a configuration plan with any structure node.

**share driver expression** In PeopleSoft Enterprise Business Planning, a named planning method similar to a driver expression, but which you can set up globally for shared use within a single planning application or to be shared between multiple planning applications through PeopleSoft Enterprise Warehouse.

**short-term customer** A customer not in the system who is entered during sales order entry using a template.


Copyright © 1992-2006, Oracle. All rights reserved.

**single signon** With single signon, users can, after being authenticated by a PeopleSoft Enterprise application server, access a second PeopleSoft Enterprise application server without entering a user ID or password.

**source key process** In PeopleSoft Enterprise Campus Solutions, a process that relates a particular transaction to the source of the charge or financial aid. On selected pages, you can drill down into particular charges.

**source transaction** In commitment control, any transaction generated in a PeopleSoft Enterprise or third-party application that is integrated with commitment control and which can be checked against commitment control budgets. For example, a pre-encumbrance, encumbrance, expenditure, recognized revenue, or collected revenue transaction.

**sourcing objective** For constraints, the option to designate whether a business rule is required (mandatory) or is only recommended (target).

**speed key** See *communication key.*

**SpeedChart** A user-defined shorthand key that designates several ChartKeys to be used for voucher entry. Percentages can optionally be related to each ChartKey in a SpeedChart definition.

**SpeedType** A code representing a combination of ChartField values. SpeedTypes simplify the entry of ChartFields commonly used together.

**staging** A method of consolidating selected partner offerings with the offerings from the enterprise's other partners.

**standard letter code** In PeopleSoft Enterprise Campus Solutions, a standard letter code used to identify each letter template available for use in mail merge functions. Every letter generated in the system must have a standard letter code identification.

**statutory account** Account required by a regulatory authority for recording and reporting financial results. In PeopleSoft Enterprise, this is equivalent to the Alternate Account (ALTACCT) ChartField.

**step** In PeopleSoft Enterprise Sales Incentive Management, a collection of sections in a plan. Each step corresponds to a step in the job run.

**storage level** In PeopleSoft Enterprise Inventory, identifies the level of a material storage location. Material storage locations are made up of a business unit, a storage area, and a storage level. You can set up to four storage levels.

**subcustomer qualifier** A value that groups customers into a division for which you can generate detailed history, aging, events, and profiles.

**Summary ChartField** You use summary ChartFields to create summary ledgers that roll up detail amounts based on specific detail values or on selected tree nodes. When detail values are summarized using tree nodes, summary ChartFields must be used in the summary ledger data record to accommodate the maximum length of a node name (20 characters).

**summary ledger** An accounting feature used primarily in allocations, inquiries, and PS/nVision reporting to store combined account balances from detail ledgers. Summary ledgers increase speed and efficiency of reporting by eliminating the need to summarize detail ledger balances each time a report is requested. Instead, detail balances are summarized in a background process according to user-specified criteria and stored on summary ledgers. The summary ledgers are then accessed directly for reporting.

**summary time period** In PeopleSoft Enterprise Business Planning, any time period (other than a base time period) that is an aggregate of other time periods, including other summary time periods and base time periods, such as quarter and year total.

Copyright © 1992-2006, Oracle. All rights reserved.

**summary tree** A tree used to roll up accounts for each type of report in summary ledgers. Summary trees enable you to define trees on trees. In a summary tree, the detail values are really nodes on a detail tree or another summary tree (known as the *basis* tree). A summary tree structure specifies the details on which the summary trees are to be built.

**syndicate** To distribute a production version of the enterprise catalog to partners.

**system function** In PeopleSoft Enterprise Receivables, an activity that defines how the system generates accounting entries for the general ledger.

**system source** The system source identifies the source of a transaction row in the database. For example, a transaction that originates in PeopleSoft Enterprise Expenses contains a system source code of BEX (Expenses Batch).

When PeopleSoft Enterprise Project Costing prices the source transaction row for billing, the system creates a new row with a system source code of PRP (Project Costing pricing), which represents the system source of the new row. System source codes can identify sources that are internal or external to the PeopleSoft Enterprise system. For example, processes that import data from Microsoft Project into PeopleSoft Enterprise applications create transaction rows with a source code of MSP (Microsoft Project).

**TableSet** A means of sharing similar sets of values in control tables, where the actual data values are different but the structure of the tables is the same.

**TableSet sharing** Shared data that is stored in many tables that are based on the same TableSets. Tables that use TableSet sharing contain the SETID field as an additional key or unique identifier.

**target currency** The value of the entry currency or currencies converted to a single currency for budget viewing and inquiry purposes.

**task** A deliverable item on the detailed sourcing plan.

**tax authority** In PeopleSoft Enterprise Campus Solutions, a user-defined element that combines a description and percentage of a tax with an account type, an item type, and a service impact.

**template** A template is HTML code associated with a web page. It defines the layout of the page and also where to get HTML for each part of the page. In PeopleSoft Enterprise, you use templates to build a page by combining HTML from a number of sources. For a PeopleSoft Enterprise portal, all templates must be registered in the portal registry, and each content reference must be assigned a template.

**tenant** In PeopleSoft Real Estate Management, an entity that leases real estate from a landlord.

**territory** In PeopleSoft Enterprise Sales Incentive Management, hierarchical relationships of business objects, including regions, products, customers, industries, and participants.

**third party** A company or vendor that has extensive PeopleSoft Enterprise product knowledge and whose products and integrations have been certified and are compatible with PeopleSoft Enterprise applications.

**tiered pricing** Enables different portions of a schedule to be priced differently from one another.

**time span** A relative period, such as year-to-date or current period, that various PeopleSoft General Ledger functions and reports can use when a rolling time frame, rather than a specific date, is required.

**total cost** In PeopleSoft Strategic Sourcing, the estimated dollar cost (sum of real price dollars and potential “soft” or non-price dollars) of a particular award approach.


Copyright © 1992-2006, Oracle. All rights reserved.

**travel group** In PeopleSoft Expenses, the organization's travel rules and polices that are associated with specific business units, departments, or employees. You must define at least one travel group when setting up the PeopleSoft Expenses travel feature. You must define and associate at least one travel group with a travel vendor.

**travel partner** In PeopleSoft Expenses, the travel vendor with which the organization has a contractual relationship.

**3C engine** Abbreviation for *Communications, Checklists, and Comments engine.* In PeopleSoft Enterprise Campus Solutions, the 3C engine enables you to automate business processes that involve additions, deletions, and updates to communications, checklists, and comments. You define events and triggers to engage the engine, which runs the mass change and processes the 3C records (for individuals or organizations) immediately and automatically from within business processes.

**3C group** Abbreviation for *Communications, Checklists, and Comments group.* In PeopleSoft Enterprise Campus Solutions, a method of assigning or restricting access privileges. A 3C group enables you to group specific communication categories, checklist codes, and comment categories. You can then assign the group inquiry-only access or update access, as appropriate.

**trace usage** In PeopleSoft Enterprise Manufacturing, enables the control of which components will be traced during the manufacturing process. Serial- and lot-controlled components can be traced. This is maintained in the Item Master record.

**transaction allocation** In PeopleSoft Enterprise Incentive Management, the process of identifying the owner of a transaction. When a raw transaction from a batch is allocated to a plan context, the transaction is duplicated in the PeopleSoft Enterprise Incentive Management transaction tables.

**transaction state** In PeopleSoft Enterprise Incentive Management, a value assigned by an incentive rule to a transaction. Transaction states enable sections to process only transactions that are at a specific stage in system processing. After being successfully processed, transactions may be promoted to the next transaction state and "picked up" by a different section for further processing.

**Translate table** A system edit table that stores codes and translate values for the miscellaneous fields in the database that do not warrant individual edit tables of their own.

**tree** The graphical hierarchy in PeopleSoft Enterprise systems that displays the relationship between all accounting units (for example, corporate divisions, projects, reporting groups, account numbers) and determines roll-up hierarchies.

**tuition lock** In PeopleSoft Enterprise Campus Solutions, a feature in the Tuition Calculation process that enables you to specify a point in a term after which students are charged a minimum (or *locked*) fee amount. Students are charged the locked fee amount even if they later drop classes and take less than the normal load level for that tuition charge.

**unclaimed transaction** In PeopleSoft Enterprise Incentive Management, a transaction that is not claimed by a node or participant after the allocation process has completed, usually due to missing or incomplete data. Unclaimed transactions may be manually assigned to the appropriate node or participant by a compensation administrator.

**universal navigation header** Every PeopleSoft Enterprise portal includes the universal navigation header, intended to appear at the top of every page as long as the user is signed on to the portal. In addition to providing access to the standard navigation buttons (like Home, Favorites, and signoff) the universal navigation header can also display a welcome message for each user.

**update access** In PeopleSoft Enterprise Campus Solutions, a type of security access that permits the user to edit and update data.

Copyright © 1992-2006, Oracle. All rights reserved.

| | |
|---|---|
| | See also *inquiry access*. |
| **user interaction object** | In PeopleSoft Enterprise Sales Incentive Management, used to define the reporting components and reports that a participant can access in his or her context. All PeopleSoft Enterprise Sales Incentive Management user interface objects and reports are registered as user interaction objects. User interaction objects can be linked to a compensation structure node through a compensation relationship object (individually or as groups). |
| **variable** | In PeopleSoft Enterprise Sales Incentive Management, the intermediate results of calculations. Variables hold the calculation results and are then inputs to other calculations. Variables can be plan variables that persist beyond the run of an engine or local variables that exist only during the processing of a section. |
| **VAT exception** | Abbreviation for *value-added tax exception.* A temporary or permanent exemption from paying VAT that is granted to an organization. This terms refers to both VAT exoneration and VAT suspension. |
| **VAT exempt** | Abbreviation for *value-added tax exempt.* Describes goods and services that are not subject to VAT. Organizations that supply exempt goods or services are unable to recover the related input VAT. This is also referred to as exempt without recovery. |
| **VAT exoneration** | Abbreviation for *value-added tax exoneration.* An organization that has been granted a permanent exemption from paying VAT due to the nature of that organization. |
| **VAT suspension** | Abbreviation for *value-added tax suspension.* An organization that has been granted a temporary exemption from paying VAT. |
| **warehouse** | A PeopleSoft Enterprise data warehouse that consists of predefined ETL maps, data warehouse tools, and DataMart definitions. |
| **weight or weighting** | In PeopleSoft Strategic Sourcing, how important the line or question is to the overall event. Weighting is used to score and analyze bids. For RFx and RFI events, weightings may or may not appear to bidders. |
| **work order** | In PeopleSoft Enterprise Services Procurement, enables an enterprise to create resource-based and deliverable-based transactions that specify the basic terms and conditions for hiring a specific service provider. When a service provider is hired, the service provider logs time or progress against the work order. |
| **worker** | A person who is part of the workforce; an employee or a contingent worker. |
| **workset** | A group of people and organizations that are linked together as a set. You can use worksets to simultaneously retrieve the data for a group of people and organizations and work with the information on a single page. |
| **worksheet** | A way of presenting data through a PeopleSoft Enterprise Business Analysis Modeler interface that enables users to do in-depth analysis using pivoting tables, charts, notes, and history information. |
| **worklist** | The automated to-do list that PeopleSoft Workflow creates. From the worklist, you can directly access the pages you need to perform the next action, and then return to the worklist for another item. |
| **XML link** | The XML Linking language enables you to insert elements into XML documents to create a links between resources. |
| **XML schema** | An XML definition that standardizes the representation of application messages, component interfaces, or business interlinks. |
| **XPI** | Abbreviation for *eXtended Process Integrator.* PeopleSoft XPI is the integration infrastructure that enables both real-time and batch communication with JD Edwards EnterpriseOne applications. |


Copyright © 1992-2006, Oracle. All rights reserved.

**yield by operation** In PeopleSoft Enterprise Manufacturing, the ability to plan the loss of a manufactured item on an operation-by-operation basis.

**zero-rated VAT** Abbreviation for *zero-rated value-added tax.* A VAT transaction with a VAT code that has a tax percent of zero. Used to track taxable VAT activity where no actual VAT amount is charged. Organizations that supply zero-rated goods and services can still recover the related input VAT. This is also referred to as exempt with recovery.

Copyright © 1992-2006, Oracle. All rights reserved.

Copyright © 1992-2006, Oracle. All rights reserved.

# Index


## A

action bundles 109
action framework
  architecture 143
  definition of 109
  overview 143
action objective
  registering 153
action type bundles
  registering 145
action types 109
  configuring 166
  creating 166
  overview 143
  registering 145
actions
  adding, editing, configuring 115
  display alert 117
  how they execute 144
Add Nodes to Chunk Rule page 185
additional documentation xviii
address verification service, using for credit card processing 102
aliases 137
application class
  example of accessing binds 172
  sample of resolving terms 172
application class code
  creating for design and runtime 170
application class implementation 128
  creating 171
  sample code 173
application fundamentals xvii
application server, configuring for credit card processing on Solaris 92
Attribute Details page 24
authorize-and-bill transactions, using 98
authorize-only transactions, using 99

## B

batch publish rules
  associating to a message 199
  characterizing 199
  pages used to assign 199
  setting up 184
Batch Publish Rules page 184, 199
bill-only transactions, using 100
binds
  application class accessing 172
business domain
  registering 161
Business Interlink Status report 36
business interlinks
  setting up for credit card processing on Solaris 91
  setting up for credit card processing on Windows NT 89
business units 185
  *See Also* chunking rules
BusUnit Mapping page 206

## C

Card Type page 82
category
  registering 161
chunking data 206, 207
Chunking Rule page 206, 225
chunking rules
  adding nodes to 208
  assigning business units to 185, 208
  assigning to a business unit 209
  assigning to a setID 210
  associating to publication rules 184
  creating 205
  defining 206
chunking tables, routing PeopleCode 213
combinable action 144
comments, submitting xxii
common elements xxii
component interfaces 179
conditions
  adding 113
  selecting terms for 114
  setting right-hand side values 115
contact information xxii
context objects
  managing 140
context variables 137
  characteristics of 137
contexts 108


Copyright © 1992-2006, Oracle. All rights reserved.

configuring 138
creating 163
managing 137
online and generic 137
overview 137
contextual implementations 132
Credit Card Authorize Bill and Credit IP 77
credit card interface
configuring the application server on Solaris 92
configuring the PeopleSoft Application Designer plug-in 92
defining connection parameters 79
defining credit card types 82
describing error codes 87
describing required parameters 98
describing unsupported input fields 104
setting transaction inputs 97
setting transaction outputs 97
setting up 77, 79
testing 83, 85
understanding 77
understanding a CyberSource integration 77
credit card interface business interlinks
setting up on Solaris 91
setting up on Windows NT 89
credit card interface CyberSource API
setting up 88
setting up on Solaris 90, 91
setting up on Windows NT 89
Credit Card Interface Installation page 79
credit card interface integrations
understanding a manual 78
understanding options 77
understanding XML 78
credit card interface Java interlink plug-in
setting up 88
setting up on Solaris 90, 92
setting up on Windows NT 89, 90
understanding 88
credit card interface services
using address verification 102
using the fraud screen 103
credit card interface transactions
using authorize-and-bill 98
using authorize-only 99
using bill-only 100
using credit 100
credit card options
manual processing 77
XML-compliant vendors 77
credit card types, defining accepted 82
credit transactions, using 100
cross rates, calculating 59
cross-references xxi
currencies
converting ad hoc amounts 61
defining 46
defining quotation methods 48
describing tables 43
understanding 43
understanding conversion factor fields 45
understanding requirements for conversion 46
understanding the visual rate 45
understanding triangulation 44
currency codes, defining 47
currency exchange calculator, using 61
currency quotation methods, defining 48
custom chunking tables 201
*See Also* chunking tables
custom operator expressions
creating 164
Customer Connection website xviii
CyberSource
processing credit cards with 77
understanding credit card processing with 77
CyberSource API
describing support requirements 88
setting up 88
setting up on Solaris 90, 91
setting up on Windows NT 89

## D

data library
components 127
definition of 108
how a value is passed 173
overview 127
setting log options 150
subject areas 108
Data Maintenance page 218
data subscription
in multilingual environment 189
in non-multilingual environment 189
data synchronization


Copyright © 1992-2006, Oracle. All rights reserved.

understanding for LDAP 8
using LDIFs 8
dataset roles
creating 65
understanding 63
dataset rules
creating 64
defining 63
understanding 63
datasets
defining mobile data distribution rules 66
understanding 63
understanding mobile data distribution 65
using mobile data distribution user rules 68
Directory Audit report 36
directory information trees, understanding 7
Directory Interface
configuring the LDAP directory 10
delivered business interlinks 8
generating LDAP directory reports 33
product requirements, recommendations 3
reviewing LDAP directory data 33
setting up LDAP directory authentication for 14
setting up mappings for 17
steps to use 8
understanding 7
understanding data mapping 7
using the configuration component 9
Directory Interface setup
setting up in PeopleSoft Application Designer 9
setting up in PeopleSoft Integration Broker 9
Directory Load page 30
display action
enabling 166
display alert action
configuring 117
display alert action type 143
Distinguished Name Details page 21
distinguished names defaults 23
distinguished names, understanding 7
DN, *See* distinguished names
DN Attribute Function - Directory Interface page 22
DN Defaults page 23
documentation
printed xviii
related xviii
updates xviii

## E

Effective Date Publish page 231
effective date publish utility
definition 223
full data publish 224
pages used to run 231
effective-dated messages 225
*See Also* full data publish
creating a publish rule definition 227
defining chunking rules 227
defining routing 229
effective-dated rows
publishing from the delay table 230
Enterprise Integration Points 181
*See Also* integration points
Entry Definition page 26
Entry Membership Rules page 28
entry membership rules, setting up 26
EO EIP Data Maint page 221
EOEC_CCI_RESPONSE message 78
EOEC_CCI_SYNC message 78
EOEC_CCI_TRANSACTION business interlink 77
EOP_RATECALC process
running 59
understanding 59
error correction
message setup and maintenance pages 217
messages 222
error handling utility
creating error correction pages 217
definition 215
row security 220
setting up 218
testing 221
workflow notification 222
error management process 215
exchange rates, accessing 57
Execute All option 121
Execute Limited Number option 121
execution options 121

Copyright © 1992-2006, Oracle. All rights reserved.

## F

File Inbound page 235
file layouts 179
flat file utility
about 233
inbound file processing 233, 237
inbound file processing pages 235
inbound file processing setup 235
inbound file processing testing 238
formatting data 132
fraud screen service, using for credit card processing 103
full data publish
creating effective-dated messages 225
creating the run control 228
defining a service operationl 226
defining connectors 226
defining nodes 226
Full Data Publish page 228
Full Table Publish Rules page 195

## G

generic contexts 137
generic implementations 132
glossary 259

## I

implementations
contextual 132
creating a PS query 174
creating an application class 171
definition 128
definition of 128
generic 132
registering 129
testing terms 134
Inbound File page 237
incremental messages
creating the message definition 230
effective-dated delay function 230
Integration Broker 180
integration points 181
*See Also* message integration points
naming standards, nouns 252
naming standards, verbs 249
setting up 183
setup pages 183

## J

Java interlink plug-in
setting up for credit card processing 88
setting up for credit card processing on Solaris 90, 92
setting up for credit card processing on Windows NT 89, 90
understanding credit card processing with 88

## L

Languages page 198
LDAP Data Interface Format files
synchronizing data using 8
LDAP data, synchronizing with PeopleSoft data 7
LDAP directory
configuring for Directory Interface 10
setting up authentication for Directory Interface 14
setting up mappings for use with Directory Interface 17
LDIF, *See* LDAP Data Interface Format files
location numbers 164

## M

managing contexts 137
manual option, processing credit cards with 77
Map Business Unit page 209
Map Details page 18
Map SetID page 210
mapping, messages 26
market rate definitions, creating 53
market rate indexes, defining 51
market rate types, defining 52
market rates
accessing exchange rate details 57
creating definitions 53
defining 50, 55
defining indexes 51
defining types 52
describing tables 43
understanding 43
message chunking
creating a chunking rule 205
overview 202
pages used to set up 202


Copyright © 1992-2006, Oracle. All rights reserved.

selecting chunking fields 203
when to use 203
message publishing
from batch programs 189
from online component 188
message record mapping
batch publish 200
full publish 196
message routing 185
messages, delivered 26
mobile data distribution
defining rules 66
understanding 63, 65
using user rules 68
Modify Connect DN - Directory Interface page 20

**N**

naming terms 164
nodes 185
*See Also* chunking rules, adding nodes to
notes xxi

**O**

online contexts 137
operands
location number 164
operator
selecting 115
operator sets
registering 159
operators
registering 158

**P**

page processor 188
pages
add nodes to chunk rule 185
batch publish rules 184, 199
BusUnit mapping 206
chunking rule 206, 225
data maintenance 218
effective date publish 231
EO EIP data maint 221
file inbound 235
full data publish 228
full table publish rules 195
inbound file 237
languages 198
map business unit 209
map SetIDs 210
quick map 208
record mapping 196
SetID mapping 207
workflow/security 220
PeopleBooks
ordering xviii
PeopleCode functions, setting up 22
PeopleCode, typographical conventions xx
PeopleSoft Active Analytics Framework
definition of 107
installation options 151
PeopleSoft Application Designer plug-in
configuring account access information for credit card processing 93
configuring for credit card processing 92
configuring proxy server support for credit card processing 96
setting parameter check logic for credit card processing 97
setting the decryption option for credit card processing 97
setting up tracing for credit card processing 96
policies
activating 118
associating to another trigger point 118
building and managing 111
copying 118
definition of 107
editing 112
prerequisites to building 111
reusing 120
policy
activating 118
policy groups 112
setting execution options 121
policy or policy group
adding 120
removing 119
reordering 120
preconditions
adding 120
prerequisites xvii
printed documentation xviii
PS query implementation
creating 174


Copyright © 1992-2006, Oracle. All rights reserved.

PS Query implementation 129
publish rules
assigning an application program to 200
associating to a message 195
characterizing 195
full table, assigning 194
languages, specifying 198
overview 193

**Q**

Quick Map page 208

**R**

rate definitions, creating 53
rate types, defining 52
RATE_DIV, understanding 45
RATE_MULT, understanding 45
reciprocal rates, calculating 59
Record Mapping page 196
Record.Field implementation 129
related documentation xviii
related languages
guidelines for messaging 187
scenarios 190
setup 186
resolution methods
registering 160
row security 220
*See Also* error handling utility

**S**

SetIDs Mapping page 207
setting execution options 121
guidelines 125
SQL object implementation 129
subject areas 108
defining 156
suggestions, submitting xxii

**T**

term codes 165
term implementations
testing 134
term properties 130
terms 259
configuration of 131
configuring 114
creating 130
exposing component buffer fields 163
managing 132
methods of an application class
resolving 172
naming 164
prompt details 131
registering 130
scope of implementation 131
user binds 130
Test Credit Card Interface - Card Entry/Display page 83
Test Credit Card Interface - Transaction page 85
TouchNet, understanding 77
triangulated rates, calculating 59
triangulation, understanding 44
trigger points
creating 165
definition of 107
enabling in a PeopleCode event 166
hierarchy 119
managing 118
registering 155
registration of 108
setting execution options 121
trigger types
registering 154
typographical conventions xx

**U**

Use Parent Execution Options option 121

**V**

visual cues xxi
visual rate, understanding 45

**W**

warnings xxi
workflow notification 222
*See Also* error handling utility
Workflow/Security page 220

**X**

XML Schema utility
examples 240
extracting table relationship 239
overview 239
XML-based integrations, understanding credit card processing with 78


Copyright © 1992-2006, Oracle. All rights reserved.

XML-based vendors
processing credit cards with 77

Copyright © 1992-2006, Oracle. All rights reserved.

Copyright © 1992-2006, Oracle. All rights reserved.