Oracle® Communications Billing and Revenue Management

Setting Up Pipeline Pricing

Release 15.2
G35871-01
January 2026

ORACLE

Oracle Communications Billing and Revenue Management Setting Up Pipeline Pricing, Release 15.2

G35871-01

Copyright © 2025, 2026, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

About This Content

1 About Creating a Price List

About Price Lists 1
What Is Rating? 3
- About Billable Events 3
- How BRM Rates a Billable Event 4
  - About Real-Time Rating 4
  - About Pipeline Batch Rating 5

About Different Types of Rates 5
- About External and Internal Events 5
- About Cycle Rates 6
  - About Cycle Forward Events 6
  - About Cycle Arrears Events 7
  - About Cycle Forward Arrears Events 7
  - How BRM Creates Cycle Events 8
  - Benefits of Using the Cycle Event Types 8
  - Allowing Cycle Fees to Be Prorated 8
- Applying Multiple Rates to the Same Event 9

Ways to Rate Events 9
- About Ratable Usage Metrics 9
  - About Applying Multiple RUMs to Rate an Event 9
- About Rating Based on Date and Time 10
- About Resources 10
  - About Rounding Resources 11
  - About Setting Rules for Resource Consumption 11
- About Rating Based on Event Attributes 11
- About Rating Based on Event or Resource Quantity 11

About Products 12
- Specifying Rates in Products 12
  - Using Products to Rate Different Services 13
  - Using Products to Charge Different Amounts for the Same Service 13
  - Associating Products with Offer Profiles 13

Copyright © 2025, 2026, Oracle and/or its affiliates.

Associating Products with Accounts 13
Using Products to Handle Complex Rating 14
Specifying Minimum Event Quantities for Rate Plans 14
Specifying How to Round Event Quantities 14
About Organizing Rates in a Product 14
About the Event Map 14
About Rate Plans 15
About Product Ownership 15
Configuring Full Day Proration 15
Specifying Which Services Are Rated by a Product 16
About Specifying the Events to Rate in a Product 16
About Discount Validity Rounding 17
Specifying Product Types 17
Specifying General Product Information 18
Defining How Many Products a Customer Can Purchase 18
Restricting When a Product Is Available 18
Specifying Product Priority 19
Rating Based on Product Provisioning 19
Providing Deal-Level Discounts 19
Defining Extended Attributes for Products 19
About Deals 20
About the Best Pricing Configuration 21
About Base Deals 21
About Alternate Deals 21
An Example of Best Pricing Calculation 22
Purchasing Deals after an Account Is Created 23
Providing Discounts with Deals 23
Prohibiting, Allowing, and Requiring Deal Modification 23
About Product and Discount Validity Periods 23
Setting the Effective Periods of Products and Discounts 23
About Effective Periods That Start on First Usage 24
About Product and Discount Status at Purchase 26
Assigning Services to Deals 26
Using Deals to Bill Customers on Demand 26
About Deal Dependencies 27
Strategies for Creating Deals 27
Using Deals Across Time Zones 27
Transitioning Between Deals 28
About Add-On Products in Deals 28
Defining Extended Attributes 28
About Plans 29
About Applying Credit Limits to Resources 29

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Credit Thresholds and Credit Floors 30
Credit Limit and Floor Options 31
Setting a Dynamic Credit Floor 31
About Effective Periods That Start on First Usage 32
Tracking Resources by Service 32
Grouping Services by Subscription 32
Creating CSR Plans 33
Using Plans to Bill Customers on Demand 33
Strategies for Creating Plans 33
Transitioning between Plans 34
Defining Extended Attributes for Plans 34
About Subscription Terms 35
About Penalty Fees 35
About Automatic Renewals 36
About Bundle Terms and Package Terms 36
About Obsoleting Subscription Terms 37
About Plan Lists 37
About the Default and CSR Plan Lists 38
About Offer Profiles 38
Price List Example 38
About Setting Up a Price List 39
Prerequisites for Creating a Price List 40
Planning Your Price List 41
About Using Pricing Center 41
Who Uses Pricing Center? 42
Example of Using Pricing Center 42
Making Changes to Your Price List 47
Deleting Products 47
Logging Changes to Price Lists 47
Ensuring Price List Consistency 48
Troubleshooting a Price List 48
Displaying Price List Elements 49
Creating a Price List with the XML Pricing Interface 49
Common Price List Solutions 49
Providing Free Hours 49
Deleting Unused Free Hours 50
Charging for Canceling a Product 50
Charging a Discounted Purchase Fee Based on Date of Purchase 50
Creating a “Third-Month Free" Rate 51
Creating Discounts Based on Usage 51
Creating Products for Administrative Events 52
Advanced Rating Features 52

Copyright © 2025, 2026, Oracle and/or its affiliates.

Charging Cycle Forward Fees in Advance 52
Example: Charging Cycle Forward Fees in Advance for Monthly Cycle Fees 53
Example: Charging Cycle Forward Fees in Advance for Quarterly Cycle Fees 53
About Charging for Administrative Events 53
About Using Product-Level Provisioning to Configure Services 53
About Remittance 54
About Custom Event Analysis 54
Setting Optional Rating Flags 54

2 Understanding the Sample Pricing Plans

About the Sample Plans 1
Purpose of the Sample Plans 1
Looking at the Sample Plans 1
Opening an IPL Price List File 2
Opening an XML Price List File 2
Descriptions of the Sample Plans 2
Plans for Internet Service Providers 2
Plan 1 – Measured Web Access with Discounts 2
Plan 2 – Unlimited Web Access with Discounts 3
Plan 3 – Unlimited Internet Service with Recurring Discounts 4
Re-Creating the Sample Plans 4
Overview of Creating a Pricing Plan 4
Plans for Internet Service Providers 5
Re-Creating Plan 1 – Measured Web Access with Discounts 6
Re-Creating Plan 2 – Unlimited Web Access with Discounts 14
Re-Creating Plan 3 – Unlimited Internet Service with Recurring Discounts 23
Descriptions of the Advanced Sample Plans 26
ASP User Profile Plan 26
Concepts Illustrated 26
Customization Required for the ASP User Profile Plan 27
Structure of the ASP User Profile Plan 27
Questions About the Plan 28
Audio/Video Plan 28
Concepts Illustrated 28
Customization Required for the Audio/Video Plan 28
Structure of the Audio/Video Plan 29
Questions About the Plan 30
Customer Service Plan 30
Concepts Illustrated 30
Customization Required for the Customer Service Plan 30
Structure of the Customer Service Plan 31

Copyright © 2025, 2026, Oracle and/or its affiliates.

Questions About the Plan 32
IP Limited Access Plan 32
Concepts Illustrated 33
Customization Required for the IP Limited Access Plan 33
Structure of the IP Limited Access Plan 33
Questions About the Plan 34
Online Articles Plan 35
Concepts Illustrated 35
Customization Required for the Online Articles Plan 35
Structure of the Online Articles Plan 36
Questions About the Plan 36
T1Access Plan 37
Concepts Illustrated 37
Customization Required for the T1 Access Plan 37
Structure of the T1 Access Plan 37
Questions About the Plan 37
Web Hosting Plan 37
Concepts Illustrated 38
Customization Required for the Web Hosting Plan 38
Structure of the Web Hosting Plan 38
Questions About the Plan 39

3 Setting Up Real-Time and Pipeline Pricing and Rating

About Configuring Price List Data 1
Setting Up Resources or Balance Elements 2
Setting Up Resources for Real-Time Rating 2
Setting Up Pipeline Manager Resources or Balance Elements 3
Defining Currencies 3
Defining a Resource or Balance Element 3
Defining Currency Exchange Rates 3
Mapping Pipeline Manager Currencies and Real-Time Rating Currencies 4
Setting Up Offer Profiles 5
Setting Up Ratable Usage Metrics (RUMs) 6
About Setting Up RUMs for Real-Time Rating 6
Creating Ratable Usage Metrics 7
Setting Up Pipeline Ratable Usage Metric (RUM) Groups 8
About Defining Units of Measurement (UoMs) 8
About Defining Ratable Usage Metrics (RUMs) 8
About Defining a RUM Group 8
Converting Units of Measurement 8
Mapping Event Types to RUMs 9

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Mapping Services 10
About Creating Map Groups 10
Mapping Service Codes and Service Classes 11
About Setting Up Service Mapping 12
Mapping Events and Services 12
Mapping an Event Type to a BRM Service Type 12
Mapping Multiple Event Types to a BRM Service Type 12
Enabling the BRM Database to Store Multiple Event Types per Service 13
Populating the Event Types in the EDR Container 13
Configuring Output Streams Based on Service Code and Event Types for iRules 13
Example Procedure to Map Multiple Event Types to a Service Type 13
Mapping Usage Classes 14
About Setting Up Usage Class Mapping 15
Mapping Usage Types 15
About Setting Up Usage Type Mapping 15
Configuring the IRL_UsageType iRule for ERAs 16
Configuring the IRL_UsageType iRule for Filter Sets 18
Configuring the IRL_UsageType iRule for Both ERAs and Filter Sets 19
Configuring the FCT_IRule Module to Run the IRL_UsageType iRule 20
Adding Customer Balance Impact Data to EDRs 20
Configuring the DAT_AccountBatch Module 20
Specifying Not to Load Closed Accounts 21
Specifying Whether to Load All Account Data 22
Specifying How Much Account Data to Retrieve on Startup 22
Configuring Charge Offer Validity Checking 22
Configuring Account Charge Offer Validity Checking for Backdated Events 22
Getting Information about Loading Accounts 23
Configuring the FCT_Account Module 23
Specifying Which Data Is Used for Identifying Accounts 24
Configuring Account ID Prefixes 25
Specifying Which Noncurrency Sub-Balances to Load on Startup 25
Configuring the Noncurrency Balance Element Validity 26
Modifying and Loading the EDR Container Description 27
Modifying EDR Container Descriptions 27
EDR Container Description Format 27
Loading EDR Container Descriptions 29
Creating the EDR Load Utility Command File 29
Before Running the EDR Load Utility 30
Starting the EDR Load Utility 30
Dropping or Upgrading Incomplete Calls after Changing the EDR Container Description 30
Discarding Incomplete Calls after Changing the EDR Container Description 31
EDR Data Available for Auditing 31

Copyright © 2025, 2026, Oracle and/or its affiliates.

Upgrading Incomplete Calls to the New Container Description 31
About Serviceless Accounts as Charge Sharing Owners 34
Mapping Subscription Services in the Pipeline Manager Database 34
Setting Up Batch Rating to Assign Items Based on Event Attributes 35
How Batch Rating Assigns Custom Bill Items 36
How Batch Rating Assigns Custom Bill Items to Events for Balance Impacts 38
Creating a Batch Rating iScript for Balance Impacts 40
Verifying Item-Tag-to-Item-Type Mapping 40

4 Configuring Resource Rounding

About Resource Rounding 1
About Rounding Rules 1
How BRM Applies Rounding 2
About Rounding and A/R Actions 4
About Rounding Billed and Unbilled Items 4
About Rounding for Specific Event Types 4
About Rounding Aggregation Counter Resources for Discounting 5
About G/L Report Rounding 5
About Rounding Modes 5
About Rounding Modes That Correct for Loss of Precision 6
When Rounding Is Not Applied 7
Configuring Resource Rounding 7
About Configuring Rounding Rules 7
Prioritizing Rounding Rules 8
About Rounding Mode Values 8
Configuring Rounding Rules 9
Configuring Rounding Rules for Aggregation Counter Resources 10
About Rounding Modes for Discount Expressions 10
Configuring to Record Rounding Differences in the G/L 10
Defining a G/L ID for Rounding Differences 11
Mapping the Rounding G/L ID to an Event 12
Configuring BRM to Record Rounding Differences 12
Setting Up Rounding in Pipeline Manager 13
About Configuring the FCT_Rounding Module 13
About Rounding Rules in Pricing 14
Avoiding Rounding Conflicts in Pipeline Manager 14
Rounding Examples 14
Correcting for Precision Loss When Rounding Down 15
Rounding Using Different Modes 16
Modifying a Rounding Rule 17

Copyright © 2025, 2026, Oracle and/or its affiliates.

5 Managing Sub-Balances

About Sub-Balances 1
About Noncurrency Sub-Balances 2
How Resources in Validity-Based Sub-Balances Are Updated 2
Configuring Time-Stamp Rounding for Validity Period Start Times 3
Configuring Time-Stamp Rounding for Cycle Grants 3
Configuring Time-Stamp Rounding for Purchase Grants 4
About Noncurrency Sub-Balances That Start on First Usage 5
About Configuring Sub-Balances 6
About Sub-Balance Contributors 6
Retrieving and Updating Sub-Balances 7
Configuring Sub-Balances 7
Editing the pin_sub_bal_contributor File 7
Running the load_pin_sub_bal_contributor Utility 8
Sub-Balance Configuration Example 9
Sub-Balances per Service Example 9
Specifying the Order in Which Resource Sub-Balances Are Consumed 10
Consumption Rule Descriptions 10
How BRM Applies Consumption Rules 12
How Batch Rating Applies Consumption Rules 12
How Real-Time Rating Applies Consumption Rules 13
Setting Resource Consumption Rules 13
Setting Consumption Rules in Your Price Plans 13
Setting Systemwide Consumption Rules for Each Resource 13
Setting the Default Consumption Rule 14
Modifying Resource Consumption Rules 15
About Rollovers 15
About Rollover Resource Sub-Balances 16
When Rollover Events Occur 17
Deleting Expired Sub-Balances 17
Rollover Example 17
About Rolling Over Resources That Expire in Midcycle 23
Prorating Rollover Resources 24
About Rolling Over Free Resources during Plan Transition 25

6 About Real-Time Rating

About Organizing Real-Time Rates in a Product 1
About Real-Time Rate Plans 1
Specifying the Rate Plan Currency 2
Specifying the Rate Plan Tax Code 2

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Rate Tiers 2
About Real-Time Rates and Balance Impacts 4
Specifying How an Event Affects an Account Balance 4
About Scaled Impacts 4
About Fixed Balance Impacts 5
Specifying Multiple Balance Impacts for a Single Rate 5
Specifying the Validity Period of Granted Resources 6
Assigning Impact Categories to Balance Impacts 6
Assigning General Ledger (G/L) IDs to Balance Impacts 6
Allowing Charges to Be Sponsored 6
One Sponsor per Product 7
About Balance Impacts That Become Valid on First Usage 7
About Setting Resource Validity Periods Based on First Usage 8
About First-Usage Start Time for Shared Resources 9
About Synchronizing First-Usage Validity of Resources in Deals 9
About Restricting Resource Validity End Time to the Product or Discount End Time 10
About Fold Events 10
Ensuring That All of an Event Is Rated 11

7 About Pipeline Rating

About Batch Rating 1
How Events Are Rated by Using Pipeline Manager 2
How an EDR Is Processed in a Pipeline 3
About the Order of Modules in a Pipeline 4
About the Oracle CDR Format 4
About EDRs 4
About EDR Containers 5
About the EDR Container Description 5
About the Container Description File 6
About Associated Records 7
How an Input File Is Represented in EDR Containers 8
How EDRs Are Used for Managing Transactions 10
About Mapping EDR Field Names and Alias Names 10
Viewing and Creating Alias Mapping for an EDR Field 11
About Function Modules 12
About Preprocessing Modules 12
About Enrichment Modules 13
About Service Mapping Modules 13
About Zoning Modules 14
About Rating Modules 14
About Discounting Modules 14

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Roaming Modules 15
About iScripts and iRules 15
How Pipeline Manager Uses BRM Data 15
How Pipeline Manager Identifies Accounts 15
How Pipeline Manager Chooses a Charge 15
How Pipeline Manager Assigns Delayed Events to Items 16
About Accounting Cycle Delay Periods 18
Configuring an Accounting Cycle Delay Period 20
About G/L IDs 21
About Mapping Balance Elements between the Pipeline Manager Database and the BRM Database 21
How Pipeline Manager Gets Historical Data 21
About Loading Pipeline-Rated Event Data 22
About Using a Single Batch Handler to Run Multiple Loading Utilities 22
About Pipeline Rating and BRM Billing 23
Function Module Dependencies 23
Data Module Dependencies 28

8 Policy-Driven Charging

About Policy-Driven Charging in BRM 1
About Setting Up Policy-Driven Charging 2
How Policy-Driven Charging Works 3
About Notifications Related to Offer Profiles Sent by BRM 4
Configuring Your BRM Environment to Support Policy-Driven Charging of Services 4
About Setting Up and Managing Offer Profiles 5
About Using New Resource IDs in Your Offer Profiles 5
Configuring Offer Profile Data 5
Managing Offer Profile Data 6
Storing Offer Profile Data 7
Retrieving Offer Profiles 7
Modifying Offer Profiles 8
Deleting Offer Profiles 8
About Configuring Resources for Tracking by BRM 8
Configuring Event Notification for Policy-Driven Charging 9
Merging the Policy-Driven Event Notification List with the BRM Event Notification List 9
Verifying That Policy-Driven Charging Event Notification List Entries Are Enabled 10
Merging Event Notification Lists 10
Loading the Updated Event Notification File into the BRM Database 10
Enabling Enterprise Applications to Handle Policy-Driven Charging Events 10
Verifying That BRM Is Integrated with Your Enterprise Applications 11
Enabling Enterprise Applications to Process Policy-Driven Charging Events 11

Copyright © 2025, 2026, Oracle and/or its affiliates.

Setting Up One Payload Configuration File 11
Specifying the Payload Configuration File to Use 11
Updating Reservation Preferences Configuration for Policy-Driven Charging 12
Updating Reservation Preferences Configuration Settings for Resources 12
Loading Reservation Preferences for Policy-Driven Charging 12
Updating the Connection Manager for Policy-Driven Charging 13
About Configuring the Required Memory for Offer Profiles Cache 13
About Referencing Offer Profile Related Variables in pin.conf Files 13
Updating the pin.conf Configuration File for Offer Profiles 14
Customizing Information Received from BRM 14
About the PCM_OP_BAL_APPLY_MULTI_BAL_IMPACTS Opcode 15
About the PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS Policy Opcode 15
How Balances Are Processed for Out-of-Session Notifications 15
How Balances Are Processed for Notifications for Prepaid Sessions 15
Customizing Information in Offer Profile Threshold Breach Notifications 16
About the PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD Opcode 16
Determining Whether Balance Impacts Trigger Notifications 16
Readjusting Quotas for Requests to Update and Authorize 17
Readjusting Quotas for Other AAA Requests 17
About the PCM_OP_TCF_AAA_GET_SUBSCRIBER_PROFILE Opcode 17
Customizing Information Received from BRM 18
About the PCM_OP_ACT_POL_EVENT_NOTIFY Policy Opcode 18
Customizing Information in Event Notifications for Policy-Driven Charging 18
Managing Offer Profiles with Opcodes 18
Managing Offer Profile Data with Price List Opcodes 18
Creating, Modifying, and Deleting Offer Profiles with PCM_OP_PRICE_SET_PRICE_LIST 18
Retrieving Offer Profiles with PCM_OP_PRICE_GET_PRICE_LIST 19
Managing Offer Profiles with Offer Profile Opcodes 19
About the PCM_OP_OFFER_PROFILE_SET_OFFER_PROFILE Opcode 19
Customizing Subscriber Preferences Data for Policy-Driven Charging 20
Customizing Business Events Associated with Policy-Driven Charging 20
Sample Threshold Breach Notification 20

9 Working with Promotions

About Promotions 1
Applying Promotions On Special Dates 1
Mapping Promotion Tags to Promotion Events 2
Applying Promotions for Special Events or Actions 3
Configuring Event Notification for Promotions 4

Copyright © 2025, 2026, Oracle and/or its affiliates.

Mapping BRM Events to Promotion Events 4

10 Working with Provisioning Tags

About Provisioning Tags 1
Defining Provisioning Tags for Telco Services by Using the pin_telco_tags file 2
Configuring Provisioning Tags in the pin_telco_tags File 2
Loading the pin_telco_tags File 4
Defining Account-Level ERAs in the pin_telco_tags File 5
Deciding Which Provisioning Tag Method to Use 5
Using the Provisioning Tag Framework 6
Configuring Provisioning Tags 6
Specifying an Opcode in a Provisioning Tag to Create an ERA 8
Variables for Parameter Values 10
Default Provisioning Tag 10
Using Custom Opcodes 11
Configuring Provisioning Tags for Policy-Driven Charging 11
About the Default Provisioning Tag Provided by BRM 11
Collecting Data for Provisioning Tags 12
Configuring Provisioning Tags for Offer Profiles 12
Modifying Provisioning Tags 13
Loading Provisioning Tag Configurations 13
Loading Provisioning Tags for Policy-Driven Charging 14
Modifying and Compiling the Provisioning Policy Source File 15
Using a Policy Source File to Set Up Provisioning 16
Sample Provisioning Tag XML File 16
Default Provisioning Tag for Policy-Driven Charging 17

11 Working with Extended Rating Attributes

About Extended Rating Attributes 1
About Multiple ERA Lists 2
About Configuring ERAs in Customer Center 2
About Sharing ERAs 3
Creating ERAs 3
Creating Friends and Family ERAs 3
Sample Friends and Family Provisioning Tag 6
Using ERAs with Multiple Lists 7
Using Model Selectors for ERAs 7
Using Charge Selectors for ERAs 8
About Rating Based on Friends and Family ERAs 8
Real-Time Rating for Friends and Family ERAs 9

Copyright © 2025, 2026, Oracle and/or its affiliates.

Customizing Real-Time Rating for Friends and Family ERAs 10
Prioritizing Rates for ERAs in Real-Time Rating 10
Pipeline Rating for Friends and Family ERAs 11
Prioritizing Rates and Discounts for ERAs in Pipeline Rating 12
Improving Pipeline Rating Performance for Events with ERAs 12
Customizing Pipeline Rating for Friends and Family ERAs 13
Customizing ERA Names and Descriptions for Client Applications 14
Creating Names and Descriptions for ERAs Defined in the Provisioning Tag Framework 15

12 Working with Price Tags

About Price Tags 1
Creating Price Tag Configuration Objects 2
Modifying Price Tag Configuration Objects 4
Using Price Tags With Sequential Discounts 5

13 Rating Implementation and Customization

How Rating Works 1
How BRM Rates and Records Usage Events 1
Specifying the Rating Mode 4
FM Rate Opcodes Called by PCM_OP_ACT_USAGE 5
Generating Ratable Usage Metrics 7
About Calculating the Maximum Available Usage 8
About Currency Conversion 8
About Applying Cycle Forward Fees 9
About Applying Cycle Arrears Fees 9
Customizing the Cycle Interval for Products 10
About Restricting the End Time of Granted Resources That Start on First Usage 11
Configuring Real-Time Rating to Restrict Resource Validity End Time 12
About Applying Folds 12
About Applying Folds Only For Account-Level Products 13
Customizing How Folds Are Applied 13
Customizing the Order to Apply Folds 13
Specifying Which Resources to Fold 14
Customizing Which Resources to Fold When Products Are Canceled 14
Assigning Rate Plans and Impact Categories to Events 14
Rating an Event Based on Extended Rating Attributes 15
Modifying Rated Events 17
Customizing How to Calculate RUMs 18
Rating and Recording Activity Events 18
Managing Sessions 19

Copyright © 2025, 2026, Oracle and/or its affiliates.

Starting Sessions 20
Recording the Start of a Session 20
Updating a Session Event 21
Recording the End of a Session 21
Rating and Recording Session Events 21
Loading Sessions in Real Time 22
Managing Price Lists 22
Committing Price List Data to the BRM Database 22
Retrieving Price List Data from the BRM Database 23
Managing Individual Pricing Objects 24
Managing /product Objects 25
Managing the Validity Period of Granted Resources 26
Modifying Rate Types 29
Customizing How to Create and Delete Products 29
Using Opcodes to Customize Products 30
Managing /discount Objects 31
Creating /discount Objects 32
Creating Pipeline Discount Models 32
Modifying /discount Objects 33
Deleting /discount Objects 33
Customizing /discount Objects 34
Configuring Discount Validity Rounding 35
Retrieving Discount Data 35
Managing /deal Objects 35
Specifying Relative Start and End Times for Products and Discounts in a Deal 37
Storing Start and End Times for Products and Discounts in a Deal 38
Customizing How to Create and Delete Deals 39
Managing /plan Objects 40
Managing /dependency Objects 41
Customizing How to Create and Delete Dependencies 41
Flist /dependency Arrays 42
Managing /transition Objects 42
Customizing How to Create and Delete Transitions 43
Managing /group/plan_list Objects 44
Managing /sponsorship Objects 44
Managing Tax Selectors and Tax Exemption Selectors 45
Creating Tax Selectors 46
Retrieving Tax Selectors 46
Retrieving the Tax Code from a Selector 47
Managing Filter Sets 47
Creating Filter Sets 47
Updating Filter Sets 48

Copyright © 2025, 2026, Oracle and/or its affiliates.

Deleting Filter Sets 48

14 Testing Your Price List

About Using a Test Database 1
Creating Test Accounts 1
Generating Test Account Activity 2
Running sample_act 2
Checking Results of Running sample_act 3
Advancing the Date 3
Setting Up the pin_virtual_time Utility 4
Running pin_virtual_time 5
Stopping and Starting BRM 5
Generating a Test Invoice 6
Example Price List Test 6

15 Using the XML Pricing Interface to Create a Price List

About Creating and Modifying Price List Files 1
About the XML Pricing Interface 1
About Downloading and Loading XML Price List Files 2
Prerequisites for the XML Pricing Interface 2
Starting the XML Pricing Interface 2
Getting Help for the XML Pricing Interface 3
Working in Interactive and Non-Interactive Mode 3
Creating Price Lists with the XML Pricing Interface 3
Modifying Existing Price Lists with the XML Pricing Interface 4
Moving a Price List between Databases with the XML Pricing Interface 5
Downloading Price Lists from the Database with the XML Pricing Interface 5
Downloading a Full Price List 5
Downloading a Subset of the Price List 6
Downloading Price Lists in Batches 7
Loading a Price List into the Database with the XML Pricing Interface 8
Purging Price Lists from the BRM Database 9
Purging the Entire Price List from the BRM Database 9
Deleting a Subset of the Price List from the BRM Database 9
Deleting Pricing Objects from the BRM Database 10
Purging Dependency or Transition Objects from the BRM Database 10

Copyright © 2025, 2026, Oracle and/or its affiliates.

16 XML Examples of Creating Pricing Components

Prorating Fees for Billing DOM Changes 1
Setting Proration for Products in a Deal 2
Configuring Subscription Terms 3
Setting Full Day Proration 5
Adding Price Tags to Products and Discounts 6
Allowing Customers to Exceed Their Credit Limit 7
Transitioning Plans and Deals 9
Configuring Add-On Products in Deals 9
Purchasing the Same Product or Discount Multiple Times 11
Aligning Recurring Charges and Product Validity to a Specific Day of the Month 13
Splitting Noncurrency Balances into Multiple Validity Periods 15
Setting Discount Validity Rounding 15
Setting Product Cycle Alignment for Reactivated Deals 16
Activating Products in Plans and Deals on First Usage 17
Using Date Ranges for Versioning 17
Defining Product Specification Attributes for Pricing Components 18
Stop Rating Inactive, Canceled, or SuspendedActive Accounts 21
Setting Loan Thresholds for Plans 22
Enabling Dynamic Credit Floors in Plans 22
Creating Tax Selectors for Products 23
Creating Tax Exemption Selectors for Products 25
Importing Tied Discount Offer Configurations into BRM 26

17 Real-Time Rating Based on Multiple RUMs

About Rating Based on Multiple RUMs 1
Applying Multiple RUMs to Rate an Event 2
Example of Mapping an Event to Multiple RUMs 2

18 Rating Based on Multiple RUMs with Pipeline Manager

About Rating Based on Multiple RUMs with Pipeline Manager 1
Applying Multiple RUMs to Rate a Batch Event 1

19 Real-Time Rating Based on Date and Time

About Rating Based on Date and Time 1
Ways to Specify When Rates Are Valid 1
Using Rate Tiers to Set Up Rating Based on Date and Time 2
    Specifying Which Rate Tiers Are Applied First 2
    Specifying How Long Rate Tiers Are Valid 2

Copyright © 2025, 2026, Oracle and/or its affiliates.

Specifying When Rate Tiers Are Valid 2
Using Date, Day, and Time Ranges Together 2
Using the Correct Time Zone 3
Specifying a Time Zone Mode 3
Resetting the Server Time Zone 3
Configuring Applications to Generate Time Zone Data 4
Using Event Start Time to Determine a Product's Validity 5
Dynamically Changing One-Time and Recurring Fees Based On the Date 5

20 Rating by Date and Time with Pipeline Manager

About Rating by Date and Time with Pipeline Manager 1
About Special Day Calendars 2
Configuring Date and Time Rating 2
About Special Day Rates 2
Setting Up Global Special Day Rates 3
Setting Up Account-Specific Special Day Rates 3

21 Real-Time Rating Based on Event or Resource Quantity

About Quantity-Based Rating 1
Selecting a Quantity Discount Bracket Based on Previous Rating 1
Selecting a Quantity Discount Bracket Based on a Resource Balance 3
Example 1: Rating Based on the Quantity of an Event 4
Example 2: Rating Based on Total Event Quantities 4
Example 3: Rating Based on Total Number of Events 5
Example 4: Rating Based on a Resource Balance Not Affected by the Rated Event 6
Example 5: Rating an Event As If Using a Single Balance Impact Group 7

22 Setting Up Zones for Batch Pipeline Rating

About Zoning 1
About Impact Categories 2
About Wholesale and Retail Impact Categories 2
About Standard Zones 3
About Geographical Zones 3
Displaying Zoning Information on an Invoice 3
About Pipeline Manager Zone Modules 4
Setting Up APN Mapping 4
About APN Maps 5
Creating APN Maps 6
Setting Up Prerating 6

Copyright © 2025, 2026, Oracle and/or its affiliates.

Creating Zone Data Files 7
Zone Data File 7
Standard Zone File 8
Geographical Zone File 8
Area Code Coordinate File 9
About Usage Scenario Mapping 9
Setting Up Usage Scenario Mapping 10
About Usage Scenario Maps 11
Creating a Usage Scenario Map File 12
Converting Zone Values for Output 14
Setting Up Multi-Segment Zoning 14
Configuring Segments in the FCT_SegZoneNoCust Module 14

23 Real-Time Rating Using Event Attributes

About Using Event Attributes to Rate Events 1
Using Event Attributes to Select Rate Plans 1
Using Event Attributes to Define Impact Categories 2
About the Default Impact Category 3
Creating Impact Categories 3
Creating Unassigned Impact Categories 4
About Charging for Custom Events and Attributes 4
Charging for Custom Events and Attributes 4
Using Custom Fields with Real-Time Rating 5
About Real-Time Zoning 5
About Setting Up Zones 6
Real-Time Zoning Pipeline Architecture 7
Setting Up Zones by Using Pricing Center 8
About Setting Up Zones by Using Zone Mapper 8
About Defining Zone Maps for Products 9
Options for Matching Event Attribute Values 9
Changing and Deleting Zone Maps 9
Selecting Multiple Rate Plans and Impact Categories 10
Customizing Zone Mapping 10
How Zone Mapping Works 10
Customizing Zone Mapping 10
Finding Zone Maps 11
Getting Zone Maps from the BRM Database 11
Saving Zone Map Data 11
Getting Zone Maps and Impact Categories from the Pipeline Manager Database 11
Using the Rate Plan Selector in Pricing Center 12
Using Wildcards to Match Event Data 12

Copyright © 2025, 2026, Oracle and/or its affiliates.

Specifying the Priority for Attributes 12
Maintaining the Order of Rate Plan Selectors 13

24 Migrating Pricing Data from Legacy Databases

About Migrating Legacy Data 1
Overview of the Migration Process 2
Legacy Data XML File Example 2
Guidelines for Mapping Legacy Data 2
Charge XML File Example 3
Mapping Legacy Data 4
About the Types of Objects to Migrate 4
Working with and Modifying the XSD 9
Before Loading Legacy Data 10
Configuring the Registry File for LoadIfwConfig Utility 10
Loading Legacy Data into the Pipeline Manager Database with LoadIfwConfig 11
Deleting Data from the Pipeline Manager Database with LoadIfwConfig 11
Deleting Data Sample XML File 11
Updating Data with the LoadIfwConfig Utility 12
Exporting Data from the Database with the LoadIfwConfig Utility 12
Troubleshooting 12

25 Improving Real-Time Rating Performance

Improving Real-Time Rating Performance 1
Changing the Precision of Rounded and Calculated Values 1
Setting the Interval for Checking for Price List Changes 1
Setting the Interval for Updating Zone Maps 2
Filtering the ERAs Considered During Rating and Discounting 2
Enabling and Disabling the Caching of Customized Products 3
Configuring the Maximum Number of Products and Discounts Cached 4
Improving Performance for Loading Large Price Lists 5

26 Pricing Utilities

load_brm_pricing 1
loadchangesets 4
load_event_map 5
load_pin_beid 7
load_pin_impact_category 8
load_pin_rum 9
load_pin_spec_rates 11

Copyright © 2025, 2026, Oracle and/or its affiliates.

load_pin_sub_bal_contributor 11
loadpricelist 13
load_usage_map 19
pin_apply_promotion 20
pin_clean_offer_override 21

Copyright © 2025, 2026, Oracle and/or its affiliates.

About This Content

This book describes how to create price lists using Oracle Communications Billing and Revenue Management Pipeline Manager.

Audience

This guide is intended for pricing administrators and charging experts.

Copyright © 2025, 2026, Oracle and/or its affiliates.

1

About Creating a Price List

This chapter presents an overview of how you can create an Oracle Communications Billing and Revenue Management (BRM) price list in Pricing Center to specify how to charge for your services.

Topics in this document:

- About Price Lists
- What Is Rating?
- About Different Types of Rates
- Ways to Rate Events
- About Products
- About Deals
- About Plans
- About Plan Lists
- About Subscription Terms
- About Offer Profiles
- Price List Example
- About Setting Up a Price List
- Making Changes to Your Price List
- Ensuring Price List Consistency
- Troubleshooting a Price List
- Displaying Price List Elements
- Creating a Price List with the XML Pricing Interface
- Common Price List Solutions
- Advanced Rating Features

Before reading this chapter, you should have a basic understanding of BRM charging. See "Overview of Charging" in *BRM Concepts*.

About Price Lists

You create a price list to define how much to charge for your services. A *price list* consists of several components:

Plans

You use *plans* to offer your services to customers. For example, if your company provides Internet access and email, your plans might include:

- A plan that offers wireless telephony.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- A plan that offers only Internet access.
- A plan that offers Internet access and email.

During account creation, your customers are presented with a list of plans (for example, “Unlimited Internet Access" or “500 free wireless minutes with unlimited roaming").

Deals

A plan consists of one or more deals. You use deals to define different ways to charge for services. For example, you can offer two different deals for Internet access:

- A deal that includes a setup fee.
- A deal that has no setup fee.

Each deal is typically associated with a specific service. For example:

- A plan that offers only Internet access includes only Internet access deals.
- A plan that offers Internet access and email includes two deals, one for Internet access and one for email.

In addition, you can purchase more than one instance of the same deal and have it applied to a single account or service.

Products

A deal consists of one or more products. You use products to package and organize the rates that define how much to charge for your services.

Your products might include:

- A setup fee.
- A monthly subscription fee.
- Usage fees for telephone calls.

  Usage fees can be rated in real time or in a batch rating pipeline.
- An offer profile

  An offer profile is associated with a product or discount.

Offer Profiles

You use *offer profiles* in association with products and discounts to provide policy-driven provisioning of services. Each offer profile consists of the following data:

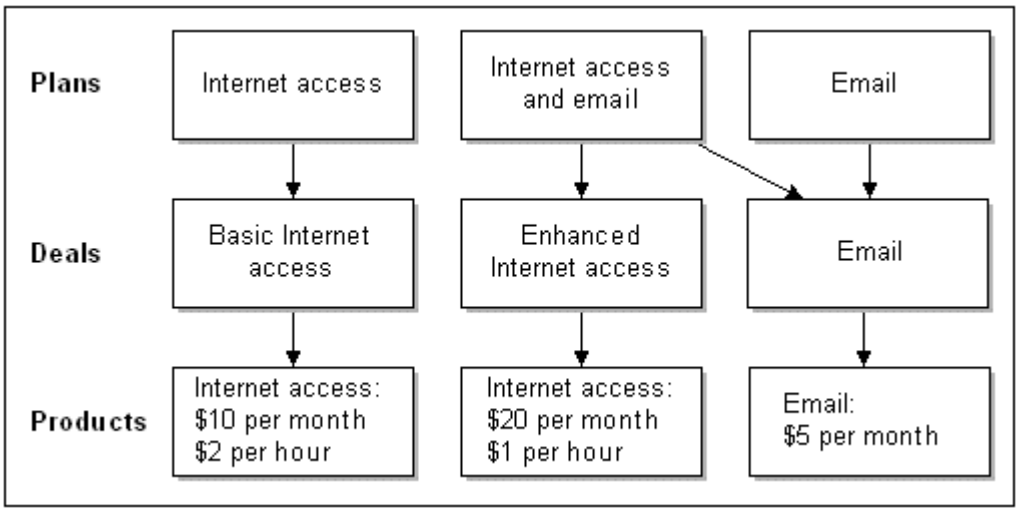
- A unique name
- An array of policy labels, with each policy label composed of rate tiers

Figure 1-1 shows how a simple price list is organized. In this example:

- The Internet access plan includes the Basic Internet access deal.
- The Email plan includes the Email deal.
- The Internet access and email plan includes the Enhanced Internet access deal and the Email deal.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 1-1 Simple Price List



In this figure, notice that you specify how much to charge for your services at the lowest level, in the rates that are included in products. Therefore, when you specify how much to charge for your services, you start by creating products and rates.

What Is Rating?

Rating is the process of measuring customer activity, determining how much to charge for it, and adding the charge to the customer's account balance.

For example, if Internet access is rated at $1 per hour:

1. A customer has a 2-hour dialup session.
2. BRM rates the session at $1 per hour.
3. The customer's account balance increases by $2.

In a typical business, thousands of customers log in daily, generating millions of interactions that must be managed and charged for. BRM manages those interactions by creating and storing billable events.

About Billable Events

An *event* is a record in the BRM database of a customer or administrative action. For example:

- When a customer logs in to a dialup session, BRM creates a session event, which stores data about the session, such as the start time and end time.
- When a customer service representative (CSR) changes a customer's password, BRM creates an activity event, which stores information such as the identification of the CSR who made the password change.

When you set up your price list, you define which events you want to charge for. These events are called *billable events*. To determine how much to charge a customer for a billable event, BRM rates the event.

Copyright © 2025, 2026, Oracle and/or its affiliates.

How BRM Rates a Billable Event

You can rate billable events in two ways:

- *Real-time rating* monitors and rates service usage as it happens, such as Internet access. Real-time rating is performed by rating opcodes.
- *Batch rating* rates events that have been recorded in files, such as telephony call detail records (CDRs). Batch rating is performed by Pipeline Manager or by Universal Event (UE) Loader. Most telco services use Pipeline Manager for batch rating.

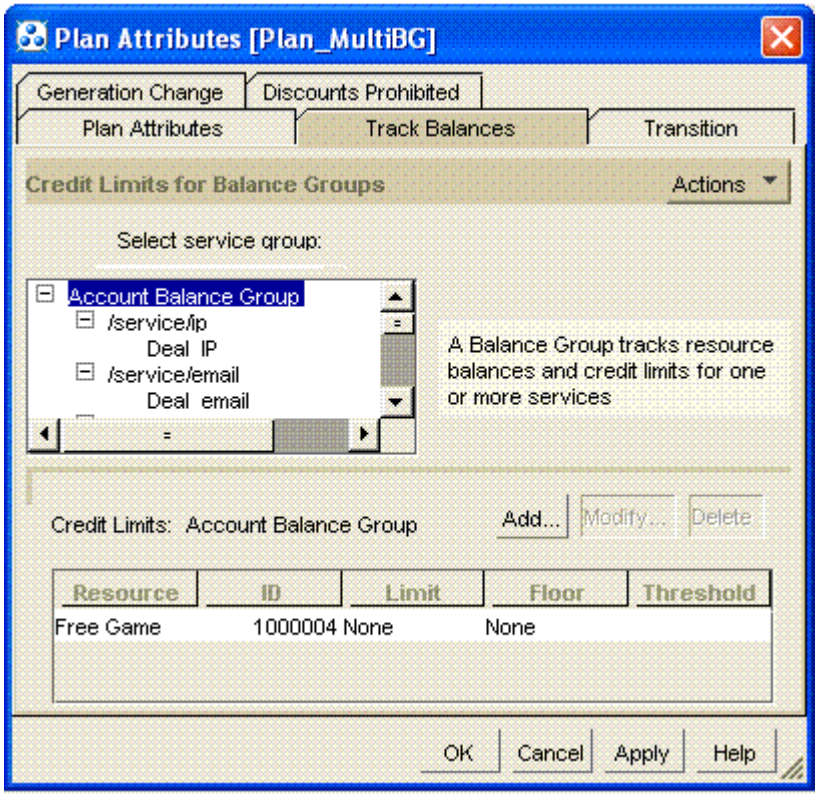
About Real-Time Rating

Real-time rating is used for services such as dial-up access and email. For example, to rate internet access in real time, BRM does the following:

1. Determines the length of the session by comparing the start time and end time.
2. Applies a charge to the amount of time. For example, if you specify to charge $1 per hour for Internet access, a 10-hour session costs $10. This charge is called the *balance impact*.
3. Adds the total charge for the event to the customer's account balance.

Figure 1-2 shows how a billable event is captured, rated, and recorded in the BRM database.

Figure 1-2 Track Balances Tab



Copyright © 2025, 2026, Oracle and/or its affiliates.

About Pipeline Batch Rating

Pipeline batch rating is commonly used for rating telephony services, where large amounts of events are recorded in files. For example, to rate telephone calls:

1. Pipeline Manager reads data from CDRs to determine the time and duration of each call.
2. BRM applies a charge based on the duration of the call. For example, if you set a rate of 10 cents per minute, a 10-minute call costs $1. This charge is called the *balance impact*.
3. BRM adds the total charge for the event to the customer's account balance.

> **Note**
>
> You can use UE Loader to import data from event log files as billable events and rate them in real-time. These event log files may include data from web servers or other services that record events in files. Once the events are loaded, BRM rates them and updates the customers' account balances accordingly. For more information, see *BRM Loading Events*.

For more information, see "About Pipeline Rating" in *BRM Setting Up Pipeline Rating and Discounting*.

About Different Types of Rates

Before you set up your price list, you must determine which types of rates to use:

- *Usage rates* rate service usage, such as telephone calls or Internet dialup sessions.
- *Cycle rates* rate recurring fees that are not generated by usage (for example, a monthly subscription fee). See "About Cycle Rates".
- *Purchase rates* charge for nonrecurring fees, such as setup fees. Purchase events are created when a customer purchases a product.
- For account and product cancellations, use **cancel events**. Cancel events occur only when a product is canceled. Cancel events are not affected by how much a customer uses a service.

  When you close an account, all the products in the account are canceled. Therefore, cancel events are generated for all the canceled products. You do not need to cancel the products first to generate cancel events.

> **Note**
>
> You use pipeline rating only for rating usage events. You can use real-time rating to rate all types of events.

About External and Internal Events

An *external* billable event is a usage event created by the customer outside of the BRM system (for example, when a customer makes a telephone call).

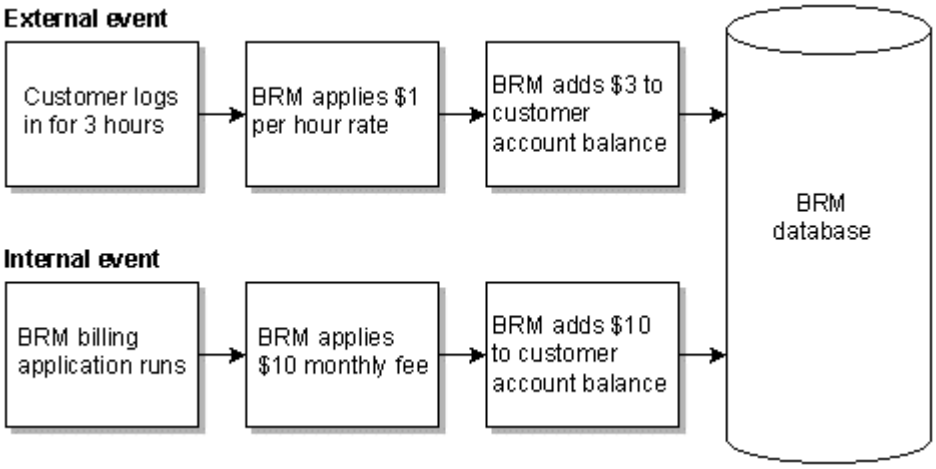
Copyright © 2025, 2026, Oracle and/or its affiliates.

An *internal* billable event is an event that is not generated by the customer, but is instead generated by the BRM system. For example, on the customer's billing day, BRM creates a monthly fee event. This event is rated by a cycle rate.

Different rates are used to rate external and internal events.

Figure 1-3 shows how BRM rates external and internal events. The external event was triggered by a dialup session. The internal event was triggered by running a billing utility.

Figure 1-3 External and Internal Event Rating Flow



About Cycle Rates

To charge subscription fees, such as monthly payments for having an account, you create rates for cycle events.

There are three types of cycle events: cycle forward, cycle arrears, and cycle forward arrears.

About Cycle Forward Events

Cycle forward events charge a fee for future service. For example, when a customer pays the monthly cycle forward fee for a cell phone service, the customer pays for the coming month. With a yearly cycle forward fee, the customer pays for the entire year in advance.

Cycle forward events typically occur at the end of a billing cycle to charge the customer for the upcoming billing cycle, but you can define your own flexible cycles.

There are five cycle forward event types to support: monthly, bimonthly, quarterly, semiannual, and annual fees. Multi-month cycle forward events can occur on any date; they do not have to be synchronized to the start or end of a month.

> **Note**
>
> You can also configure support for cycles of any length (for example, weekly or every five months).

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Cycle Arrears Events

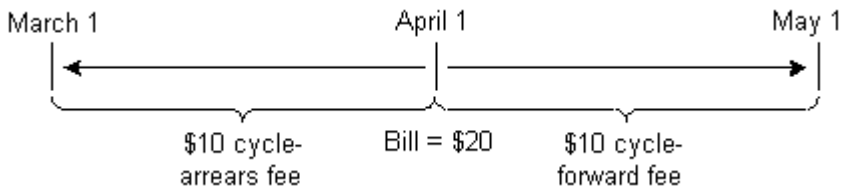
Cycle arrears events occur at the end of each month to charge customers for the previous month's services. When paying a cycle arrears fee, customers are settling for the month that has passed.

> **Note**
>
> BRM does not support multimonth cycle arrears fees.

For example, in Figure 1-4, a customer has two plans, each with a cycle fee. The bill generated on April 1 includes:

- A $10 cycle arrears fee for March
- A $10 cycle forward fee for April

Figure 1-4 Cycle Arrears and Cycle Forward Fees



About Cycle Forward Arrears Events

Cycle forward arrears events occur at the beginning of the month to charge customers for the upcoming month. However, the cycle fees are not billed until the end of the month when billing is run. When a customer pays a cycle-forward arrears fee, they are paying for the month that has just passed.

> **Note**
>
> BRM does not support multimonth cycle arrears fees.

BRM adds cycle forward arrears fees to account balances as unbilled revenue at the start of the cycle. This allows for the recognition of unbilled cycle arrears fees when running G/L reports prior to the cycle's end. For more information, see "About Collecting General Ledger Data" in *BRM Collecting General Ledger Data*.

Cycle forward arrears fees are stored in cycle forward arrears items. The cycle forward arrears event is assigned to an item that belongs to the next accounting cycle. This way, the fee is tracked in the account balance for the current cycle, but it is not billed until the end of that cycle.

About Using Delayed Billing

Delayed billing is important because cycle-forward arrears fees are assigned to items belonging to the next billing cycle.

Copyright © 2025, 2026, Oracle and/or its affiliates.

With delayed billing, BRM tracks events in the current cycle that are billed in the next accounting cycle. When the bill item for the next accounting cycle is created, BRM assigns the tracked events to that bill item.

If a delayed billing period is unnecessary, set the value of the delayed period to **0**.

For information about configuring delayed billing, see "Setting Up Delayed Billing" in *BRM Configuring and Running Billing*.

How BRM Creates Cycle Events

Cycle events are internal events created when you run the **pin_bill_accts** billing utility. For example, if the **pin_bill_accts** utility finds that an account's billing date is due, and the customer has signed up for a plan that includes a monthly cycle forward fee, BRM creates a Monthly Cycle Forward Event.

For information about accounting cycles, see "Managing Billing and Accounting Cycles" in *BRM Configuring and Running Billing*.

Benefits of Using the Cycle Event Types

The advantage of using a cycle arrears event is that all of the usage and cycle fees for a particular month are included in the same bill.

The advantage of using a cycle forward event is that you collect payments sooner. Most implementations use cycle forward events.

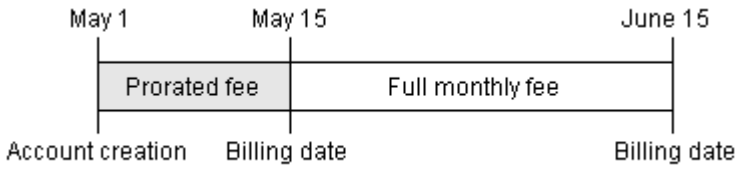
The advantage of using a cycle forward arrears event is that all of the usage and cycle fees for a particular month are included in the same bill, *and* the cycle fee is recorded in the G/L when a G/L report is run before the fee is billed.

Allowing Cycle Fees to Be Prorated

Cycle events are not related to how much a customer uses a service. However, you can enable cycle charges to be prorated. By default, BRM prorates cycle fees in the following cases:

- When a customer registers for an account and the account billing day is not the same day that the account was created. In this case, the customer is billed for the partial month that falls between registration and the first billing date as shown in Figure 1-5.

Figure 1-5 Prorating Cycle Fees



In this example, a prorated cycle forward fee is charged on May 1, when the account is created to pay for account ownership from May 1 through May 15.

- When a customer changes the billing day of month (DOM) and a partial month is created as a result.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- When a customer cancels a product before the end of the billing cycle.

You can specify if a cycle fee should be prorated, if the complete fee should be applied, or if no fee should be applied.

When you specify that a fee can be prorated, you specify which resources can be prorated. For example, you can prorate currency, but not free internet access hours.

For details on proration, see "About Proration" in *BRM Configuring and Running Billing*.

Applying Multiple Rates to the Same Event

You can apply multiple rates to the same event by creating multiple rates within a rate plan. For example:

- You can apply different time of day rates to a single type of dial-up event. See:
  - Real-Time Rating Based on Date and Time
  - Rating by Date and Time with Pipeline Manager
- You can apply a different rate based on the quantity that has already been rated. See "Real-Time Rating Based on Event or Resource Quantity" for more information.
- If you are defining rates and creating your price list using the XML Pricing Interface, you can apply a different rate based on event date, purchase date, or service instantiation date. This lets you use different rates to create different versions of the same product. See "Using Date Ranges for Versioning" for more information.

Ways to Rate Events

When you create your price list, you can use many different ways to define rates (for example, which data you measure, how to measure the data (duration or occurrence), and time of day).

About Ratable Usage Metrics

You can rate an event based on any data captured in the event. For example, you can measure and rate how long a session is or measure and rate the number of bytes downloaded. The event data that you use to rate an event is called the *ratable usage metric*, or RUM. Common RUMs are:

- Duration. You rate based on how long a usage event was.
- Occurrence. You rate based on how many events occurred, independent of their duration.

You set up RUMs for pipeline batch rating and real-time rating. For information, see the following:

- Pipeline batch rating: "Setting Up Real-Time and Pipeline Pricing and Rating".
- Real-time rating: "About Setting Up RUMs for Real-Time Rating".

About Applying Multiple RUMs to Rate an Event

By default, BRM rates an event by using a single RUM that is specified in the product's usage map. You can also set up your products to rate an event based on multiple RUMs.

For example, you can set up a product to rate fax events based on the following RUMs:

- The number of bytes transmitted

Copyright © 2025, 2026, Oracle and/or its affiliates.

- The number of pages transmitted
- The duration of the fax session

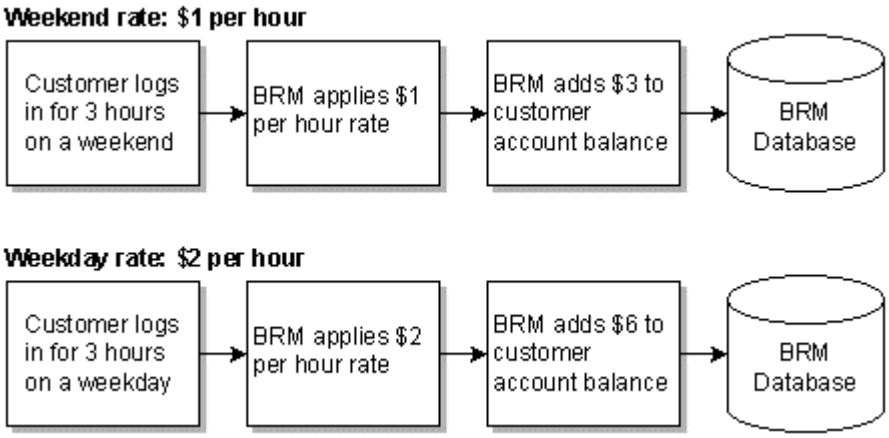
For more information, see "Real-Time Rating Based on Multiple RUMs".

About Rating Based on Date and Time

You can use different rates depending on the date and time that the event occurred. For example, you can apply one rate on weekdays and another rate on weekends.

Figure 1-6 shows how the same event can be rated differently based on when it occurred.

Figure 1-6 Rating Based on Date and Time



You can use pipeline batch rating and real-time rating to rate events based on date and time. For information, see the following:

- Pipeline batch rating: "Rating by Date and Time with Pipeline Manager".
- Real-time rating: "Real-Time Rating Based on Date and Time".

About Resources

When you create rates, you can specify different resources to use for the balance impact. A resource is an asset of economic value, such as US dollars and free Internet access hours.

- You must create currency resources for the system currency and the account currency that you support. For example, if you have Canadian customers, you must create the Canadian dollar resource.
- You must create noncurrency resources if your rates include noncurrency balance impacts such as free minutes or Internet access hours. For example, when you provide free Internet hours, you need a resource to create balance impacts for the access hours.

You set up resources for pipeline batch rating and real-time rating. For information, see the following:

- Pipeline batch rating: "Rating by Date and Time with Pipeline Manager".
- Real-time rating: "Real-Time Rating Based on Date and Time".

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Rounding Resources

You round resources for various reasons (for example, to increase the accuracy of rating and discounting, display rounded amounts on bills, and show the correct decimal value for different types of currencies such as dollars and yen).

BRM enables you to round resources based on the type of resource or currency, the type of event such as purchase, usage, and discount events, and the process that performs the rounding such as rating, discounting, and billing. These options enable you to set up rounding in specific ways. For example, you can round up to a precision of six decimal places for rating usage events, round down for discounting those usage events, and round to the nearest two decimal places when billing the usage.

You set up rounding rules for pipeline batch rating and real-time rating. For information, see the PDC Online Help or Pricing Center Help.

About Setting Rules for Resource Consumption

You can specify the order in which resource sub-balances are consumed. For example, if a customer has several groups of free minutes that expire at different times, you use consumption rules to indicate which minutes to use first, based on the validity period start time and end time.

You set up consumption rules for pipeline batch rating and real-time rating. For information, see "Specifying the Order in Which Resource Sub-Balances Are Consumed".

About Rating Based on Event Attributes

With attribute-based rating, you can specify different event attributes to consider for rating. For example:

- Telephony events include two event attributes that record the call origin and destination. If you offer a telephony service, you can rate call events according to call origin and destination.
- Telephony events also include an event attribute that records the type of call (for example, a call made with a calling card). You can use this attribute to rate calls made with a calling card differently from other calls.

You can use pipeline batch rating and real-time rating to rate events based on event attributes. For information, see the following:

- Pipeline batch rating: "Rating by Date and Time with Pipeline Manager".
- Real-time rating: "Real-Time Rating Based on Date and Time".

About Rating Based on Event or Resource Quantity

You can rate events based on the quantity that has already been rated. For example:

- 10 cents per minute for the first 30 minutes of usage.
- 6 cents per minute for the next 60 minutes of usage.
- 4 cents per minute for usage over 90 minutes.

You can use pipeline batch rating and real-time rating to rate events based on event or resource quantity. For information, see the following:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Pipeline batch rating: "Rating by Date and Time with Pipeline Manager".
- Real-time rating: "Real-Time Rating Based on Date and Time".

About Products

When you create your price list, you organize rates into products. A *product* is the basic unit of your price list. Each product defines a set of events, usually associated with a service, and the balance impacts that are applied when those events occur.

Specifying Rates in Products

When you create a product, you specify basic information about the product, such as the name and description. You also specify the following rating information:

- The events to rate (for example, monthly cycle forward or IP Dialup events). You can rate more than one type of event in a product.

> **Note**
>
> A single product cannot include multiple cycle events that have the same frequency and type of balance impact, such as a cycle fee. For example, if you add a monthly cycle forward event to a product, you cannot also add a monthly cycle arrears or monthly cycle forward arrears event to the same product. Instead, Oracle recommends that you create separate products for each cycle forward event.

- The service that the product applies to. For example, to create a product for rating an IP Dialup service, you use **/service/ip**.
- For each event, the ratable usage metric (RUM). For example:
  - For cycle forward events, rate by occurrence.
  - For dialup events, rate by duration.
- For each event, the resource used for the balance impact. For example:
  - To charge money, the resource might be US Dollars.
  - To give free hours, the resource might be Dialup hours.
- The balance impact of each event (for example, $1 per hour).

For example, a product for rating an IP Dialup service might include the rating information shown in Table 1-1.

Table 1-1 Rating Information for Dialup Service

Event: Monthly Cycle Forward Event	Event: IP Dialup Event
• RUM: Occurrence • Resource: US Dollars • Balance impact: $20 per occurrence	• RUM: Duration • Resource: US Dollars • Balance impact: $1 per hour

In addition to the rating information shown in this example, you can supply additional information, such as how to calculate taxes, the dates and times that a rate is valid, how to rate based on quantity, and if a rate is discountable.

Copyright © 2025, 2026, Oracle and/or its affiliates.

You can create products in many different ways to accommodate many pricing scenarios.

Using Products to Rate Different Services

When you offer multiple services, you typically create products for each service. Table 1-2 shows an example.

Table 1-2 Products and Services

IP Access Product	Email Product	Fax Product
$10 purchase fee	No purchase fee	$10 purchase fee
$10 monthly charge	$5 monthly charge	$5 monthly charge
$1 per hour usage fee	$5 per month for an extra mailbox	No usage fee

You can create products for different services in a single plan, but a product cannot be used to rate more than one service.

Using Products to Charge Different Amounts for the Same Service

You can use products to charge for the same service in different ways. You can do this by rating combinations of internal and external events. Table 1-3 shows an example.

Table 1-3 Rating Combinations

Standard Product	Low Usage Product	Unlimited Usage Product
$20 purchase fee	$10 purchase fee	$20 purchase fee
$10 monthly charge	$5 monthly charge	$20 monthly charge
$1 per hour usage fee	$2 per hour usage fee	No usage fee

Associating Products with Offer Profiles

To provide policy-driven provisioning of services, you associate products with offer profiles. For example, if you create an offer profile called *Data Pro Unlimited*, you use that name as the provisioning tag for your product. Resource tracking is made possible when you configure the resource id for the resource named in the offer profile (for example, **100009** for *Megabytes Used*) as the resource counter in the product.

Associating Products with Accounts

You can create products that associate rates with accounts. For example, you can create a product that includes monthly charges for an account independent of the services that a customer owns as shown in Table 1-4.

Table 1-4 Associating Products and Accounts

Account Product	IP Access Product	Email Product
No purchase fee	$10 purchase fee	$10 purchase fee
$10 monthly charge	No monthly charge	No monthly charge
$5 cancel fee	$1 per hour usage fee	No usage fee

Copyright © 2025, 2026, Oracle and/or its affiliates.

Using Products to Handle Complex Rating

You can rate a single event in different ways in the same product. Table 1-5 shows an example.

Table 1-5 Complex Rating with Products

Simple Product	Complex Product
$10 purchase fee	Purchase fees: • $10 purchase fee if purchased after December • $5 purchase fee if purchased after October • No purchase fee if purchased before or during October
$1 per hour usage fee	Usage fees: • $1 per hour on weekends • $2 per hour on weekdays • First 20 weekend hours free • First 10 weekday hours free

Specifying Minimum Event Quantities for Rate Plans

You can specify a minimum event quantity to rate. For example, suppose the minimum quantity for the IP access rate is 120 seconds (2 minutes). IP access events of less than 120 seconds are rated as if they were 120 seconds long.

Specifying How to Round Event Quantities

You can specify how to round fractional event quantities. For example, if you round Internet access to the nearest second, and a customer connects to the Internet for 2 minutes and 39.76 seconds, the event is rated at 2 minutes and 40 seconds. Event quantities are always rounded up.

You can take advantage of event quantity rounding to charge for consistent blocks of time (for example, 5 cents for every 10 seconds of an IP telephony call).

> **Note**
>
> Event quantity rounding is different from resource rounding. With resource rounding, you round the amount to charge. With event quantity rounding, you round the amount of usage. See "Setting Up Resources for Real-Time Rating".

About Organizing Rates in a Product

You organize rates in a product in a hierarchical structure. This enables you to create multiple components at each level; for example, you can rate multiple events in a single product and create multiple rates for each event.

About the Event Map

The event map specifies the events that are being rated in the product. For example:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Monthly Cycle Forward Event
- IP Dialup Event

See "Mapping Events and Services".

About Rate Plans

You use rate plans to define how much to charge for an event. There are two separate but related kinds of rate plans. In the simplest terms, real-time rate plans are used for real-time rating and pipeline rate plans are used for batch rating.

> **Note**
>
> If you use Pipeline Manager to rate events, you must use the same rate plan names in your price list and in the pipeline pricing configuration.

About Product Ownership

During account creation, the customer chooses a plan. A plan is a package of deals, each of which in turn is a package of products. Therefore, what the customer actually owns is not a plan, but a set of products.

If you create different products for the same service, different customers might use the same service, but pay different charges, based on the products that they own.

For example, you might have two customer accounts that use the same service, but have different products as shown in Table 1-6.

Table 1-6 Products and Accounts

Account 1	Account 2
IP product 1: • Service: IP Dialup • Cycle forward event: $20 • Usage events: $2	IP product 2: • Service: IP Dialup • Cycle forward event: $10 • Usage events: $1

When BRM rates events for these customers, the charges for Account 1 are different from the charges for Account 2, even though the services and the events being rated are the same type.

> **Note**
>
> When you manage a customer's account, you manage their products. For example, you can change the product status or customize the product's pricing. When creating your price list, consider how products will be managed after purchase.

Configuring Full Day Proration

When you create a charge offer, set the rounding values for the following:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- **The validity period**: You can specify that the validity period starts at the purchase time or midnight of the day the product is purchased. Alternatively, you can set to use the systemwide setting in the CM **pin.conf** file.
- **The charging scale**: You can specify whether to charge for a full day or a partial day for the first day of the billing cycle. Alternatively, you can set to use the validity period setting.

  See "Setting Full Day Proration".

Specifying Which Services Are Rated by a Product

To define which services are rated by a product, you specify whether a product is associated with an *account* or with a *single service*.

- **Account**. This product applies only to an account. For example, you could charge a monthly fee just for having an account, regardless of the services. Even if all services are inactivated, the monthly charge would still be applied. You can also use an account-level product for an account sign-up fee.

  You might create account-level products if you offer several services that customers do not own for any length of time (for example, access to Web simulcasts). In that case, a customer might own the following products:

  – Account-level product: $20 monthly fee
  – IP Dialup service product: $10 per occurrence
- **Single service**. Usually, you associate a product with a service (for example, IP Dialup or Email). When you do this, the rates in the product are applied to that service only. It is easier to manage a price list when each product has a purchase level associated with a single service.

  When a product applies only to a single service, the product is inactivated when the service is inactivated, and charges for the service stop.

  You can associate as many products with the same service as you want.

> **Note**
>
> Products must be associated with the same service as the deals that contain them. For example, if your product allows **/service/ip**, any deals containing that product must also be valid for **/service/ip**.

About Specifying the Events to Rate in a Product

You might need to charge different types of fees for a service, such as subscription fees and usage fees. When you create a product for a service, you define the types of fees to charge by specifying which events to rate. The group of events rated by a single product is called an *event map*.

For example, to charge purchase, subscription, and usage fees for a service, a product's event map includes:

- Product Purchase Fee Event
- Monthly Cycle Forward Event
- IP Dialup Event

Copyright © 2025, 2026, Oracle and/or its affiliates.

You do not have to include all types of events in a single product. A product requires only one event in the event map, so you could create individual products for each type of event that you want to rate.

How you choose which events are rated in a product depends in large part on the services you offer, but it also depends on how you manage your business and your customers. For example, you might change your pricing often, provide sponsored services, or provide numerous types of discounts. Before you finalize your price plan, run some discounting and customer management scenarios to determine if your products are organized in a way that supports your business.

About Discount Validity Rounding

You can specify whether a discount offer's validity period starts when the discount is purchased (such as 15:00) or at midnight (00:00) of the purchase date.

You can also specify to apply the discount to the entire day, even when rounding to midnight is disabled. For example, a discount purchased at 15:00 on June 1 would be valid from 00:00 through 24:00 on June 1.

In BRM, if the value of the field is **NOT_SET** in the **/discount** object, the value of the business profile key (DISCOUNT_VALIDITY_ROUNDING) overrides the value at the discount level. You must load the **pin_business_profile.xml** file into the database by using the **load_pin_business_profile** utility.

The PIN_FLD_FLAGS field in **/purchased_discount** is set to one of the following:

- **0x100**: When discount validity rounding is ON
- **0x200**: When discount validity rounding is OFF

Specifying Product Types

There are three types of products: item, subscription, and system.

- *Item products* contain rates that are applied only once (for example, a purchase fee). The only event type you can use for an item product is the purchase event.
- *Subscription products* contain rates that are applied on an ongoing basis (for example, usage charges and monthly charges).
- *System products* contain rates that can be applied to all products in your price list. For example, you could create products to charge for IP usage with various limitations for various customers. You could then create a system product to charge a default usage rate of $0.10 per minute for all your customers when those other products are not valid.

You specify product types when you use Pricing Center to create a product as shown in Figure 1-7.

Figure 1-7 Product Type in Pricing Center

Copyright © 2025, 2026, Oracle and/or its affiliates.

Specifying General Product Information

To create a product, you specify a product name and description. In addition, you specify information about provisioning, taxes, priority, and time- and quantity-based validity ranges.

If you are using the XML pricing interface to create your price list, you can also define what happens if a customer purchases the same product more than once. By default it is purchased as a new product, but you can specify that it instead:

- Replaces the existing product
- Extends the existing product if it is purchased within a specified grace period after the existing product expires

See "Purchasing the Same Product or Discount Multiple Times" for more information.

Defining How Many Products a Customer Can Purchase

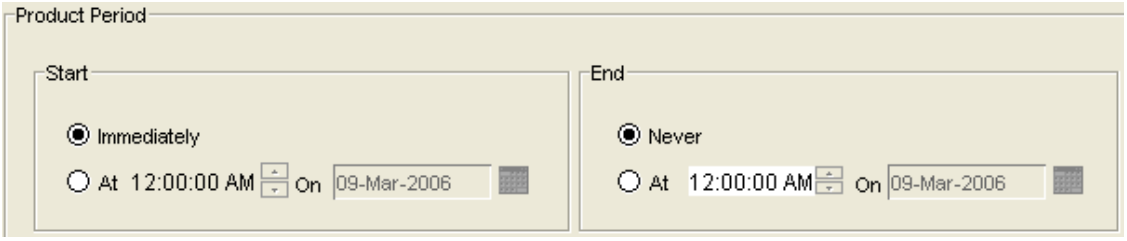
You can define the following quantity attributes:

- You can specify whether a customer can purchase part of a product for a reduced price. For example, if an IP fax product provides 100 pages faxed for $50, you can allow customers to purchase 50 pages for $25.
- You can define the minimum and maximum number of products that can be purchased. For example, if a product includes an item such as a t-shirt, you might want to limit the number of t-shirts that can be purchased simultaneously.
- You can define the minimum and maximum number of products that can be owned at any given time. For example, if a product provides an email service, you might want to limit the number of email login names that a single customer can own.

Restricting When a Product Is Available

You can specify that a product is always valid, valid from a future date forward, valid until a future date, or valid for some definite period in the future as shown in Figure 1-8.

Figure 1-8 Restricting Product Availability



For example, if you accept the default value, as shown above, your product is available immediately and always.

You can also specify:

- When each rate is valid. See "Real-Time Rating Based on Date and Time".
- When products and their fees are effective for the accounts that purchase them. See "About Product and Discount Validity Periods".

Copyright © 2025, 2026, Oracle and/or its affiliates.

Specifying Product Priority

When more than one product applies to an event, BRM considers products in the order of product priority. You set product priority when you create a product.

For information on enabling product priority while applying cycle fee, see the discussion about enabling product priority while applying cycle fee in *BRM Configuring and Running Billing*.

Rating Based on Product Provisioning

Product provisioning enables you to define how a service is configured and to rate each configuration accordingly. To activate provisioning, you select a provisioning tag.

For more information, see "About Using Product-Level Provisioning to Configure Services".

Providing Deal-Level Discounts

You use deals to specify discount amounts. You use balance impacts to specify that the resource in the appropriate balance impact is discountable.

> **Note**
>
> These discounts apply only to events rated by real-time rating. You can create sophisticated discounts that apply to both events rated by real-time rating and events rated by the batch pipeline.

For example, you might have a product that rates two cycle events. For one event you give free hours and for the other you charge a monthly fee. You must create a deal that discounts the monthly fee. However, when you provide a discount in a deal, the discount applies to all cycle forward fees; you cannot choose which cycle fee to discount. To discount only the monthly fee, you make the resource for the monthly fee discountable and the resource used to give free hours non-discountable.

The results of a discount can be calculated two ways. By default, the discount is applied to the product of the rate and quantity:

[(rate * quantity) – discount]

If you prefer, you can apply the discount to the rate itself:

[(rate – discount) * quantity]

See "Setting Optional Rating Flags" for instructions about setting this option.

Defining Extended Attributes for Products

When you create your price list using the XML Pricing Interface, you can define extended attributes for products. Extended attributes store information used by external applications, such as enterprise product catalogs, beyond what is provided in BRM. See "Defining Extended Attributes" for more information.

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Deals

A deal is a set of products. Deals are typically used for the following purposes:

- To package a set of related products.
- To provide discounts on products for promotional purposes.
- To define start and end dates for a product. The start and end dates that a deal provides must be within the valid start and end dates of the product.
- To allow customers to purchase more than one of the same product with only one transaction.
- To enable on-demand billing.

For example, you can offer two different deals for Internet access:

- A deal that includes a $15 setup fee.
- A deal that has no setup fee.

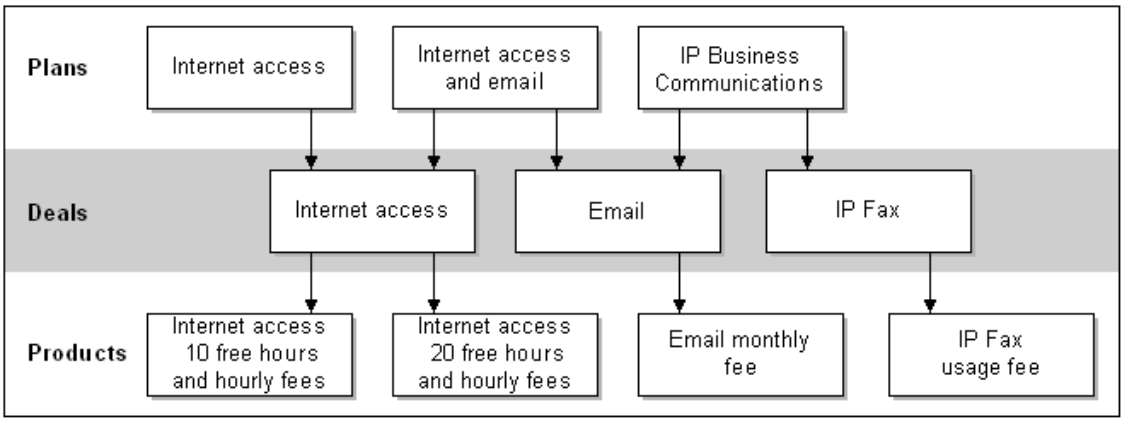
Each deal is typically associated with a specific service. For example:

- A plan that offers only Internet access includes only Internet access deals.
- A plan that offers Internet access and email includes two deals, one for Internet access and one for email.
- A plan that offers email and IP fax service includes two deals, one for email and one for IP fax.

One deal can contain any number of products, and two different deals can contain the same product. Therefore, packaging products in deals adds flexibility to your pricing structure without requiring you to create additional products. Figure 1-9 shows a sample offering consisting of related plans, deals, and products.

As with products, you can define the valid dates for deal availability, and the purchase level of the deal.

Figure 1-9 Deal Relationships to Plans and Products



If you are using the XML Pricing Interface to create your price list, you can also:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Define what happens if a customer purchases the same product or discount in a deal more than once. By default, it is purchased as a new product or discount, but you can specify that it instead replaces the existing product or discount, or extends the existing product. See "Purchasing the Same Product or Discount Multiple Times" for more information.
- Define how the cycle for products align if a customer suspends the deal and later reactivates it. By default, the cycle aligns with either the billing date or the original purchase date, but you can specify that it instead aligns with the reactivation date. See "Setting Product Cycle Alignment for Reactivated Deals" for more information.
- Define how each product in a deal calculates prorated cycle fees. By default, cycle fees are prorated based on the actual number of days in a month, such as 28, 30, or 31, but you can specify to calculate prorated cycle fees based on a 30-day month, regardless of the number of days in the month. See "Setting Proration for Products in a Deal" for more information.

About the Best Pricing Configuration

Best pricing is a pricing configuration that consists of a base deal and a set of alternate deals that are used to compare charges with the base deal to arrive at the lowest rate for the customer. For more information, see "About Base Deals" and "About Alternate Deals".

Because a deal is associated with a service, best pricing calculation is performed at the service level. You can calculate best pricing for multiple services by grouping services into subscription groups. For more information, see the discussion about how BRM calculates the best price for subscription groups in *BRM Configuring and Running Billing*.

> **Note**
>
> If the best pricing deal is purchased at the account level, best pricing calculation includes all the events from all the service instances of the account. The products and discounts from all services are included in rating.

About Base Deals

A *base deal* is a deal that is used to rate events as they occur. For each base deal, you can set up a set of alternate deals for the best pricing configuration.

At billing time, the charges calculated using the base deal are compared with the charges of each alternate deal to arrive at the best price.

> **Note**
>
> You can configure only one base deal in each best pricing configuration.

About Alternate Deals

An *alternate deal* is a deal that is used to rate the events at billing time for comparing the resulting charges with the charges calculated using the base deal. BRM performs a calc-only rerating operation using the alternate deals at billing time. You can also calculate the best price at any time during the billing cycle by calling the PCM_OP_SUBSCRIPTION_CALC_BEST_PRICING opcode in calc-only mode or by using

Copyright © 2025, 2026, Oracle and/or its affiliates.

Customer Center. You can then charge your customer the lowest price calculated using the deal that costs the least.

> **Note**
>
> The number of alternate deals in your configuration affects performance of the best pricing calculation.

For more information, see the discussion about offering the best price to your customers in *BRM Configuring and Running Billing*.

For each alternate deal, you can specify:

- A minimum amount that is charged for the deal whether the services are used or not. For example, the minimum amount can be the sum of the cycle fees applicable for the deal.
- A set of conditions that must be met for the alternate deal to qualify for best pricing calculation.

> **Note**
>
> The minimum charge and the conditions limit the number of alternate deals that are considered for the best pricing calculation and thereby improve the performance of the best pricing calculation.

An alternate deal can have additional charges that are not in the base deal, and the rates in the alternate deal override the rates in the base deal.

An Example of Best Pricing Calculation

Consider an account that has used a total of 700 minutes and 10 Short Message Service (SMS) messages in the month and has the best pricing configuration shown in Table 1-7.

Table 1-7 Best Pricing Calculation Example

Base Deal	Alternate Deal
Minutes usage charge = $0.05	Minutes usage charge = $0.03
SMS usage = $0.10	Free SMS usage
NA	Cycle fees = $10

The charges calculated are shown in Table 1-8.

Table 1-8 Calculating Charges

Usage Type	Base Deal Charges	Alternate Deal Charges
Minutes	700 x 0.05 = $35.00	700 x 0.03 = $21.00
SMS	10 x 0.10 = $1.00	0
Cycle fee	0	$10
Total	**35 + 1 + 0 = $36**	**21 + 0 +10 = $31**

Copyright © 2025, 2026, Oracle and/or its affiliates.

In this example, the alternate deal charges are lower than the base deal charges, so the alternate deal is the best deal and is used for rating and billing the customer.

Purchasing Deals after an Account Is Created

When a customer registers for an account, the customer purchases deals contained in a plan. However, once an account is created, the customer can purchase add-on deals that are not included in a plan, but the deal must be associated with the same service.

Providing Discounts with Deals

You use deals to provide discounts on products. You can provide separate discounts for each type of rate. For example, if the product contains a cycle rate (for a monthly fee) and a usage rate, you can discount either or both rates.

For each rate category in each product, you can specify a percentage to discount and the dates the discount applies to.

When configuring discounts, you can:

- Specify if rates are discountable. See "Providing Deal-Level Discounts".
- Provide fractional discounts up to two decimal places (for example, 10.25%, but not 10.255%).
- Define what happens if a customer purchases the same discount more than once, if you are using the XML pricing interface to create your price list. By default, the new purchase is treated as a new, separate discount, but you can specify that it instead replaces the existing discount. See "Purchasing the Same Product or Discount Multiple Times" for more information.

Prohibiting, Allowing, and Requiring Deal Modification

You can also give discounts to products in Customer Center (for example, fixed or percentage discounts on monthly fees). When you create a deal, you can specify whether to prohibit, allow, or require deal modification.

For example, to set a price based on individual customer input, you can specify that a deal must be modified.

About Product and Discount Validity Periods

Products and discounts have the following validity periods:

- The validity period that defines when a product or discount is generally available for purchase. You define this validity period when creating products and discounts. See "Restricting When a Product Is Available".
- The validity periods that define when the product and discount are effective for the accounts that purchase them and when the product's fees begin to accrue in the account balance and to be discounted. You define these validity periods when adding products and discounts to deals. See "Setting the Effective Periods of Products and Discounts".

Setting the Effective Periods of Products and Discounts

You specify the effective period of products and discounts by defining when the purchase, cycle, and usage periods start and end:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- The purchase period defines when the product or discount is activated and the customer can begin to use the product's service or benefit from the discount. The product's purchase start time is also the earliest time that the product's fees can begin to accumulate in the account balance.
- The cycle period defines when the cycle fees are charged or discounted.
- The usage period defines when the usage fees are charged or discounted.

The cycle and usage periods must fall within the purchase period.

You can set the purchase, cycle, and usage periods to start:

- **Immediately**: The effective period starts as soon as the customer purchases the product or discount.
- **On first usage**: The effective period starts when the product or discount is first used. The product fees are not charged until the customer uses a service in the product for the first time: for example, by making a phone call. For more information, see "About Effective Periods That Start on First Usage" for more information.
- **Relative to the product or discount purchase time**: The purchase time is the time when a product or discount is added to the account.
  - When the purchase period has a relative start time, the purchase fee is charged when the product is purchased, but the customer cannot use the product's service or benefit from the discount until the relative period ends.
  - When a product's cycle or usage period has a relative start time, the cycle or usage fees are not charged (or rated) until the relative period ends, even if the service has been activated. This option enables you to waive subscription or usage fees for a period.
  - When a discount's cycle or usage period has a relative start time, the discount is not applied to cycle or usage fees until the relative period ends.
- **On a specific day of the month**: If you use the XML pricing interface to create your price lists, you can also align purchase, cycle, and usage periods to a specific day of the month. See "Aligning Recurring Charges and Product Validity to a Specific Day of the Month" for more information.

You can set the purchase, cycle, and usage periods to end:

- **Never**: Once activated, the product or discount is effective indefinitely. The product's fees can be charged or discounted indefinitely.
- **Relative to the start time**: The start time is when the product or discount is activated.
  - When the purchase period ends relative to the purchase start time, the product or discount is effective for the relative period specified. After the relative period ends, the customer can no longer use the product's service or benefit from the discount.
  - When a product's cycle or usage period has a relative end time, the cycle or usage fees are no longer charged when the relative period ends.
  - When a discount's cycle or usage period has a relative end time, the cycle or usage fees are no longer discounted when the relative period ends.

About Effective Periods That Start on First Usage

Setting products and discounts to start when they are first used enables you to delay charging customers for the services they purchase until they start using those services or to delay activating discounts until they can be applied to customers' usage. This is useful when you offer limited-time services or discounts that expire relative to when they are activated.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> Products that start on first usage must include usage fees. If you set a product that has no usage fee to start on first usage, the product is never activated.

You can also set up individual resources granted by products and discounts to start when the resource balances are first consumed. For more information, see "About Balance Impacts That Become Valid on First Usage".

The effective period start time is set to the start time of the event that first uses the service or triggers the discount. The end time is set based on the end time that you configure for the product or discount.

> **Note**
>
> If you use the XML pricing interface to create your price lists, you can specify that all products and discounts in a deal are activated when one of the products or deals is activated.
>
> For example, assume a deal includes products A and B and discount C. If the first service a customer uses is in product B on 15 May, all three (products A and B and discount C) are activated on 15 May. See "Activating Products in Plans and Deals on First Usage" for information and examples.

About Activating First-Usage Discounts

Discounts that start on first usage can be activated when the customer first uses a service, when the first cycle fee is applied, or when the account's bill is generated, depending on which fees are discounted and when the discount is triggered. For example, a first-usage discount on SMS messaging is activated when the customer sends the first SMS message, a first-usage discount on cycle fees is activated when the first cycle fee is applied, and a first-usage billing-time discount is activated when the first bill is generated for the account.

A discount on a particular service may not be activated when a customer first uses that service. For example, if long-distance calls are discountable for a telephony service, a customer may make multiple local calls, which do not trigger the discount's effective period, before making a long-distance call.

There are some cases in which a discount's balance impact is negated by a second discount. If this occurs, the first discount's validity period is still set.

For example, a customer purchases a plan that includes discount A and discount B:

- Discount A is configured to start when first used, gives 10% off of all calls, and has a higher priority, so it is applied first.
- Discount B is configured to start immediately and makes all birthday calls free.

If a customer makes a first-usage call on his birthday and is charged $5.00 for the call, discount A is applied first, which reduces the balance by $.50, and discount A's validity period is set. Then discount B backs out all charges. In this case, the validity period of discount A remains set.

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Setting First-Usage Validity During Pipeline Rating

If you use Pipeline Manager to rate usage, configure these pipelines to set the effective periods of products and discounts when they are first used:

- Batch rating pipeline. See "Configuring Pipeline Output for First-Usage Products, Discounts, and Balance Elements" in *BRM Setting Up Pipeline Rating and Discounting*.
- Batch discounting pipeline. See "About Setting the Validity of Balance Elements Impacted by Discounts" in *BRM Setting Up Pipeline Rating and Discounting*.
- Real-time rerating pipeline. See "Configuring the Real-Time Rerating Pipeline to Set Charge Offer Validity Periods" in *BRM Rerating Pipeline Events*.

About Product and Discount Status at Purchase

When you add a product or discount to a deal, you also specify whether the product or discount is active or inactive at the time of purchase. Additionally, for products or discounts with inactive status, you must also specify a reason code that further describes the reason for the inactive status.

You define reason codes in the **reasons.***locale* file. For more information, see "Localizing and Customizing Strings" in *BRM Developer's Guide*.

Assigning Services to Deals

You define the services that a deal applies to by assigning a purchase level to a deal. The purchase level can be any of the following:

- **A single service**. Usually, you set a deal's purchase level to a single service. The products in the deal and the rates they contain can only be applied to that service. This makes it easier to create plans: you have more flexibility in putting together a combination of deals when each deal applies to a specific service. Also, when a CSR adds a service to an account, it is easy to find the correct deals to offer when the deals are associated with individual services.
- **All accounts/no services**. When you set the purchase level to all accounts/no services, you create an *account-level deal*. Each account can have only one account-level deal. The account-level deal typically includes account-level products (for example, a product that specifies the monthly charge for owning an account, regardless of the services).

Using Deals to Bill Customers on Demand

On-demand billing enables you to bill a customer immediately for a purchase, even if the customer's billing cycle has not ended.

When you create a deal, you can flag it for on-demand billing. When a customer purchases a deal that is flagged for on-demand billing, a bill is generated immediately for the purchase fees associated with the deal.

> **Note**
>
> On-demand billing works with purchase fees only, not with cycle, usage, or cancel fees.

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Deal Dependencies

You can define dependencies between deals that set up the following relationships:

- **Prerequisites**. Specifies that an account must own a particular deal to be able to purchase an additional deal.

  > **Note**
  >
  > – A prerequisite can contain deals of different services. For example, to own a GPRS deal, an account must own a GSM deal.
  >
  > – A prerequisite deal cannot contain item products. When you create a plan with alternate deals, and if the base deal contains an item product, the purchase fails.

- **Plan requirements**. Specifies whether deals are optional or required for plans. Required deals must be purchased when a plan is purchased, whereas optional deals can be added at any time.
- **Mutual exclusivity**. Sets up a mutually exclusive relationship between two deals so that if an account owns one deal, it cannot own the other.
- **Allowed transitions**. Specifies which deals or plans can serve as replacements for others.

  Transitions specify the deals that customers can switch to and remain fully provisioned. While transitioning from one deal to another, your customers retain their devices, such as phone numbers and services.

  Customers owning deals associated with a primary service may transition to other deals associated with that primary service. The list of deals displayed as available for transition are all associated with a specific primary service.

Strategies for Creating Deals

You typically use deals for providing discounts. This enables you to create fewer products and to create a baseline of standard products that you can manipulate with deals.

For example, you could create several products, each with different rates. Or, you could create a single product and change the rates by providing discounts in deals.

Also, to make your deals easier to handle in Customer Center, you should associate each deal with a single service. A common exception is a deal that applies to all accounts, which you associate with accounts instead of a service.

When you create a deal, you can specify whether to prohibit, allow, or require deal modification. In addition, when you create a plan with options, you can specify whether the deals contained in the plan are required or optional when the plan is purchased.

Using Deals Across Time Zones

When you create a deal, BRM sets the deal's start and end dates to the current date at midnight GMT (Greenwich Mean Time). For example, if a user in the US creates a deal at any time on January 15, the system stores it as January 15 at midnight GMT. If the BRM server is located in London, the deal's time is set to GMT London time, 5 hours later than the US-based user's computer. If the US-based user opens a deal using **Modify Product**, the deal's start

Copyright © 2025, 2026, Oracle and/or its affiliates.

time is posted as 5 hours earlier than midnight on January 15 (or 7:00 PM on January 14). The start date is then recorded as January 14 rather than January 15.

To avoid a time discrepancy, add the following string to the **Infranet.properties** file:

```
infranet.user.timezone = time_zone
```

where *time_zone* is the appropriate time zone, such as **Europe/London** or **America/Los_Angeles**.

Transitioning Between Deals

You can perform three types of transitions between deals: upgrade, downgrade, and generation change. When you create deals, you can define the deals that can be transitioned.

For generation change, you can transition customers between 2G (second generation) and 3G (third generation) wireless deals and services. Deals are called 2G or 3G depending on whether they have a second- or third-generation service as their primary service type. For more information about defining generation change transitions, see "Defining a Generation Change for Packages" in *BRM Managing Customers*.

> **Note**
>
> For service-level extended rating attributes (ERAs), be aware that ERA profile information is not automatically transferred between deals during deal transition or generation change. If the two deals have some common provisioning tags, the ERA profile information can be reconfigured in the new deal.

If you create price lists using the XML Pricing Interface, you can also define how to apply cycle rates when customers transition from one deal to another in the middle of their billing cycle. By default, the cycle rates of both deals are prorated, but you can specify instead to apply only the original deal's cycle rate or only the new deal's cycle rate. See "Transitioning Plans and Deals" for more information.

About Add-On Products in Deals

When you create your price list using the XML Pricing Interface, you can include add-on products in your deals.

All products in deals are base products by default, which means they can be purchased without any prerequisites. Add-on products can be purchased only if the customer owns a valid base product.

When you create an add-on product, you also specify how to determine its validity start date. The add-on product's validity start date is the end date of a product that you specify. For example, assume product A has a validity period from June 1 through June 15. If you specify to align add-on product B's validity period with product A, product B's validity start date would be June 15. See "Configuring Add-On Products in Deals" for information.

Defining Extended Attributes

When you create your price list using the XML Pricing Interface, you can define extended attributes for products, deals, and plans. Extended attributes store information used by external

Copyright © 2025, 2026, Oracle and/or its affiliates.

applications, such as enterprise product catalogs, beyond what is provided in BRM. See "Defining Product Specification Attributes for Pricing Components" for more information.

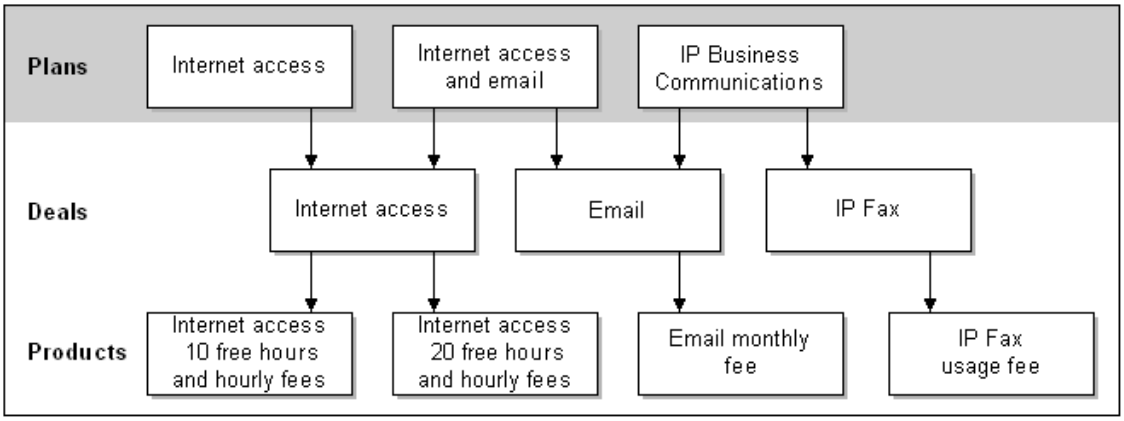
About Plans

You use *plans* to offer your services to customers. For example, if your company sells Internet access, email, and IP fax service, your plans might include:

- A plan that sells only Internet access.
- A plan that sells Internet access and email.
- A plan that sells email and IP fax service.

Figure 1-10 shows how you can include deals, and the services they apply to, in a variety of plans.

Figure 1-10 Plan Relationships to Deals and Products



Usually, plans include services. By including services in plans, a customer creates a service login name and password during registration. The only time you might not include a service in a plan is if the plan contains only account-level products and deals. In that case, you register a customer, and assume that services are added later.

One plan can contain any number of deals, and two different plans can share the same deal. By grouping deals into plans, you simplify the choices presented to customers.

> **Note**
>
> Plans always include deals. The only exception is the CSR plan.

About Applying Credit Limits to Resources

You use plans to apply credit limits to resources.

A *credit limit* is the maximum amount of a resource, such as currency or hours, that can accumulate in an account balance before the customer is prevented from logging in or using a service. For example, you might set a credit limit of $100 for an Internet access plan.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> The default credit limit for currency resource is NULL and noncurrency resource is 0. The default currency resource must have a positive value and the noncurrency resource must have a negative value.

Credit limits are defined in the plans that customers purchase. Credit limits must also be defined in the input flist when the customer account is created.

- If the credit limit is not defined in the input flist, the credit limit defaults to NULL even if the credit limit is defined in the plan.
- If the credit limit defined in the plan differs from the credit limit in the input flist, the credit limit in the input flist is used.

You use Customer Center to change the credit limit.

Though the credit limits are defined in the plans that customers purchase, the same credit limits must be defined in the input flist when the customer account is created.

About Credit Thresholds and Credit Floors

You set credit thresholds to notify customers when they are approaching the credit limit of a resource. Credit thresholds are defined in plans.

The *credit threshold* specifies the balance total that triggers an alert to the customer. You can specify the threshold in two ways:

- As a fixed value, such as $100 or 30 minutes.
- As a percentage of the credit limit, such as 90%. For example, if the credit limit is $100 and the threshold is 90%, the threshold amount is reached when the customer has a balance of $90; that is, when the customer has used 90% of the resource.

The *credit floor* is the starting point for the credit thresholds and is the lowest number that the resource value can be; that is, the number that represents no use of the resource. For currency resources, the credit floor is 0.

For noncurrency resources, such as prepaid hours, you can use a negative number for the credit floor. For example, if you give 100 prepaid hours and you set the credit limit to **0**; when the credit limit is reached, the customer has no hours remaining and cannot use the service.

You can also set the plan to use a dynamic credit floor. In this case, the credit floor is determined by the granted amounts from the sub-balances that are valid for the current cycle for the resource (for example, minutes).

The threshold works the same way, regardless of whether the credit floor is fixed or dynamic.

To notify the customer when there are only 10 hours left, you set the credit threshold and floor as follows:

- Set the credit floor to -100. This is the number that indicates none of the resource has been used.
- Set the credit threshold to 90%.

The threshold is reached at 90% of -100 hours; that is, when the customer has 10 prepaid hours left.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> You can adjust a customer's credit limit.

The credit threshold is triggered both when the balance increases and when it decreases.

You can enable BRM to provide credit threshold breach notifications as in-session notifications appended to the responses it sends to network connectivity applications. BRM sends these notifications in its responses to the authorization and reauthorization requests that BRM receives from these applications during prepaid sessions. For more information on in-session notifications in prepaid sessions, see the discussion about providing in-session notifications for network connectivity applications in *BRM Telco Integration*.

You can customize BRM to perform different actions in each case. For example, if the credit threshold is crossed when the balance is increasing, service could be turned off. When the threshold is crossed when the balance is decreasing, service could be restored.

Credit Limit and Floor Options

There are two optional settings that affect the way credit limits and floors are handled during rating:

- You can turn off the checking of credit floors. Credit floor checking sometimes causes certain events to be skipped during rating.

  With credit floor checking enabled, you might receive a threshold notification that indicates a customer has crossed the credit floor threshold, even if a system discount would have prevented the threshold from being crossed. For example, assume the credit floor is 0, the system discount is 20%, and the customer has a $20 prepaid balance. If the customer has charges for items worth $22.00, the threshold notification is generated, even though the discount reduces the charges to $17.60 (so the threshold is not actually crossed).
- You can choose whether to impose a zero credit limit if there is a balance impact to a resource in an account where the balance element for that resource is missing. If you choose to use a zero credit limit, the balance for that resource cannot exceed zero. If you choose not to use the credit limit, the balance can be any amount.

You set optional rating features using the ExtraRateFlags business parameter. See "Setting Optional Rating Flags" for more information.

Setting a Dynamic Credit Floor

Instead of setting a static credit floor, you can configure BRM to generate a dynamic credit floor based on the granted amounts from sub-balances that are valid for the current cycle for the resource (for example, minutes).

You can set a dynamic credit floor using either the **loadpricelist** utility or the PCM_OP_PRICE_SET_PRICE_LIST opcode. Additionally, you can configure the plan to use a dynamic credit floor with the PCM_OP_BILL_SET_LIMIT_AND_CR opcode. See "How BRM Handles Consumption Rules and Credit Limits" in *BRM Opcode Guide* for more information.

To set a dynamic credit floor using the **loadpricelist** utility, include the **dynamic_floor** tag with a value of **1** in the XML file processed by **loadpricelist**.

To set a dynamic credit floor using the PCM_OP_PRICE_SET_PRICE_LIST opcode, include the PIN_FLD_DYNAMIC_CREDIT_FLOOR field in the PIN_FLD_LIMIT array and set its value to **1**.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Regardless of the method used, if the dynamic credit floor is not set, the default value of **0** is used.

See "Enabling Dynamic Credit Floors in Plans" for more information about setting this field.

About Effective Periods That Start on First Usage

Setting products and discounts to start when they are first used enables you to delay charging customers for the services they purchase until they start using those services or to delay activating discounts until they can be applied to customers' usage. This is useful when you offer limited-time services or discounts that expire relative to when they are activated.

> **Note**
>
> Products that start on first usage must include usage fees. If you set a product that has no usage fee to start on first usage, the product is never activated.

If you use the XML pricing interface to create your price lists, you can set all products and discounts in a plan to be activated when one of the products or deals is activated.

For example, assume a plan includes deal 1 with products A and B, and deal 2 with product C and discount D. If the first service a customer uses is in product B on 15 May, all four (products A, B, and C, and discount D) are activated on 15 May. See "Activating Products in Plans and Deals on First Usage" for information and examples.

Tracking Resources by Service

Accounts can have multiple balance groups. By default, accounts are created with one balance group. You can add additional balance groups to track resources for specific services when you create your plans in Pricing Center.

For example, to track resources for a $50.00 prepaid wireless service, you create a balance group for the service in the plan and specify a credit limit of $0.00. You specify $0.00 because prepaid services have a credit balance, represented by a negative number, in this case, -$50.00. As customers use the service, the balance is debited until it reaches $0.00.

If you do not create additional balance groups, resources for every service a customer purchases share the same balance group. This means that resources such as free minutes are shared among all services. By creating a balance group for each service, you can control the allocation and consumption of resources for each service instance.

When customers purchase plans, the associated balance groups are added to the customers' accounts. Customers' account balances are then displayed in Customer Center for each service or set of services stored in a balance group.

CSRs can also specify credit limits for the services in a balance group when they create customer accounts. For example, customers might want to limit the monthly amount they spend on phone calls. CSRs set credit limits in Customer Center.

Grouping Services by Subscription

You can group services by subscription (for example, a set of services associated with a wireless connection). You group subscription services to track balances and bill customers for individual subscriptions rather than for the accounts.

Copyright © 2025, 2026, Oracle and/or its affiliates.

An account can contain multiple subscriptions. To group subscription services, you define the service that represents the subscription, then associate it with the services that customers actually use. For example, you might use a telco service to represent the subscription and associate it with telephony and messaging services. When customers purchase the telco subscription and use the associated services, the service fees are tracked and stored at the subscription level.

Creating CSR Plans

In addition to providing services to your customers, plans provide services to CSRs.

To create a CSR plan:

1. Use Pricing Center to create a plan that has the following attributes:
   - Use the **admin_client** service. This service provides access to Customer Center users.
   - Include no deals.
2. Add the CSR plan to the **CSR - new** plan list.

CSR plans serve two purposes:

- They control access to Customer Center.
- They allow customer management events to be recorded, including information about the CSR who generated the event. This information can be helpful when researching customer complaints.

Using Plans to Bill Customers on Demand

On-demand billing enables you to bill a customer immediately for a purchase, even if the customer's billing cycle has not ended.

When you create a plan, you can flag it for on-demand billing. When a customer purchases a plan that is flagged for on-demand billing, a bill is generated immediately for the purchase fees associated with the plan.

> **Note**
>
> On-demand billing works with purchase fees only, not with cycle, usage, or cancel fees.

For more information about on-demand billing, see the discussion about on-demand billing in *BRM Configuring and Running Billing*.

Strategies for Creating Plans

When your customers register for accounts, they are presented with a list of your plans. Similarly, CSRs see a list of plans when creating accounts. Therefore, you should be careful about your plan names and descriptions.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> A plan name can include a maximum of 255 characters. A plan description can include a maximum of 1023 characters.

You might consider using optional deals when creating plans. This enables you to include all relevant deals in one plan, without requiring customers to purchase all of them. Optional deals are available for purchase at registration time or any time afterward. For example, say a company offers a GSM service plan that includes a standard GSM deal, voice mail deal, and a text messaging deal. CSRs are required to purchase the base GSM deal for the service and have the option of adding the voice mail deal and text message deal for the customer at a later time. Adding optional deals to plans also filters the deal list in Customer Center, so that only the deals within the plan are available when CSRs add on deals to a current account.

Another strategy is to consider creating plans that form a logical upgrade path. For example, one plan might offer a per page rate for IP Fax service, while the higher-grade plan offers unlimited IP Fax service.

A third strategy is to create plan-to-plan upgrades, downgrades, or generation changes. This enables you to easily transition customers from one plan to the next by defining rules that govern how plans can be purchased.

Transitioning between Plans

You can perform three types of transitions between plans: *upgrade*, *downgrade*, and generation change. You define the plans that can be transitioned when creating plans in Pricing Center.

For generation change, you can transition customers between 2G (second generation) and 3G (third generation) wireless plans and services. Plans are called 2G or 3G depending on whether they have a second- or third-generation service as their primary service type. For more information about defining generation change transitions, see "Defining a Generation Change for Packages" in *BRM Managing Customers*.

> **Note**
>
> For service-level extended rating attributes (ERAs), be aware that ERA profile information is not automatically transferred between plans during plan transition or generation change. If the two plans have some common provisioning tags, the ERA profile information can be reconfigured in the new plan.

If you create pricing plans using XML and the **loadpricelist** utility, you can also define how to apply cycle rates when customers transition from one plan to another in the middle of their billing cycle. By default, the cycle rates of both plans are prorated, but you can specify instead to apply only the original plan's cycle rate or only the new plan's cycle rate. See "Transitioning Plans and Deals" for more information.

Defining Extended Attributes for Plans

When you create your price list using the XML Pricing Interface, you can define extended attributes for plans. Extended attributes store information used by external applications, such

Copyright © 2025, 2026, Oracle and/or its affiliates.

as enterprise product catalogs, beyond what is provided in BRM. See "Defining Product Specification Attributes for Pricing Components" for more information.

About Subscription Terms

Subscription terms specify a contract's commitment period, such as 1 month, 1 year, or 2 years. They also specify whether customers:

- Can cancel their contract before the commitment period ends
- Are charged a penalty for canceling early, such as an early termination fee, the contract balance, or a subset of the contract balance
- Have a grace period, during which the penalty is waived
- Have their contracts renewed automatically when the commitment period ends

For example, a term might specify a 1-year commitment period, allow early cancellation with a $250 early termination fee, and renew automatically with the same terms.

BRM includes standard terms that you can use for your contracts or use as examples when creating your own terms.

After you create subscription terms, you can associate them with your plans and deals.

You create subscription terms using the XML Pricing Interface only. For a list of the XML entries used to create a subscription term, see "Configuring Subscription Terms".

About Penalty Fees

You can charge your customers a penalty fee if they cancel their contract early. You can use this fee to recoup the cost of equipment or discounts given to customers at the start of a contract; for example, recoup the cost of a cell phone that was provided for free with a wireless service subscription.

The penalty fee can consist of an early termination fee, the remaining contract balance, or both. Table 1-9 describes the types of charges you can apply.

Table 1-9 Charge Types for Early Cancellation

Charge Type	Description
Early Termination Fee	The fee can be either: • A flat fee, such as 100 US dollars or 75 euros. • A fee that is reduced in equal amounts over the contract's term. For example, in a 1-year commitment period with an early termination fee of $60, you could reduce the fee by $5 each month. A customer canceling after 1 month would pay $55, and a customer canceling after 8 months would pay $20.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 1-9 (Cont.) Charge Types for Early Cancellation

Charge Type	Description
Contract Balance	The contract balance can be either: • The full contract's remaining balance. For example, assume a subscription term has a 1-year commitment period with a $50 monthly charge. Customers would be charged $50 for each month remaining in the 12-month contract. A customer canceling after 7 months would pay $250, and a customer canceling after 9 months would pay $150. • (15.0.1 or later) The remaining balance for a specified minimum contract length. For example, assume a subscription term has a 1-year commitment period, a minimum contract length of 4 months, and a $50 monthly fee. Customers would be charged $50 for each month remaining in the 4-month minimum contract length. A customer canceling after 1 month would pay $150, and a customer canceling after 3 months would pay $50. A customer canceling after 6 months would pay $0 because they already paid the 4-month minimum.

You can also add a grace period, allowing your customers to cancel a contract within the first few days or weeks after purchase without incurring a penalty fee.

About Automatic Renewals

You can set contracts to renew automatically at the end of the commitment period with the same terms or with different terms. For example, you can create a subscription term that automatically renews with the same 1-year commitment or with a different monthly commitment.

When a contract is renewed, time-based discounts, such as a first-year discount, are not renewed. Time-based discounts that exceed the contract commitment period, such as a two-year discount, are still in effect, but the end date for the discount remains the same.

Your customers can renew their contracts manually at any time, regardless of the commitment period, through your service representative or your customer care application.

About Bundle Terms and Package Terms

After you create subscription terms, you can associate them with your packages and bundles. A package's terms apply to all required bundles in the package, but any optional bundles are governed by their own, independent terms.

For example, assume you have a package that has a $20 early termination fee. That package includes a required bundle for digital textbooks and an optional bundle for online tutoring. If a customer cancels the digital textbook service after the first month, they must pay the $20 fee and the entire package is canceled, including the online tutoring service. But the terms for the online tutoring service might let the customer cancel it at any time without paying an early termination fee or canceling the rest of the package.

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Obsoleting Subscription Terms

> **Note**
>
> When a plan is obsoleted, it does not obsolete the term that is associated with it. You must obsolete the subscription term separately.

When you obsolete a subscription term, it does not impact your customers' existing contracts. Those contracts continue to be governed by the same terms. Your new product offerings, however, can no longer include the obsoleted term.

You cannot obsolete a subscription term when:

- A plan and the deals associated with the plan all use the same subscription term. Before you can obsolete the term, you must change the plan and its deals to use a different subscription term.
- The term is set to automatically renew at the end of its commitment period. Before you can obsolete the term, you must change its automatic renewal setting to **No**.

About Plan Lists

A *plan list* is a group of plans, usually offered to a single type of customer. You use plan lists to group rate plans based on customer types, such as the customer's age and address. For example, you might have the following plan lists:

- A plan list that includes plans for customers above a certain age.
- A plan list that includes plans for customers in a particular location (for example, Canadian customers).
- A plan list that includes promotional discounts that you offer for a limited time.

Grouping plans into plan lists enables you to:

- **Offer plans to customers based on customer type**. For example, in Pricing Center you can define a Gold - Senior plan list with two plans, GS1 and GS2. Based on the age group of the user, you can use the Gold - Senior plan list to limit the offerings to your customers through your customer management application to GS1 and GS2.

  > **Note**
  >
  > You must maintain mapping between your plan lists and your customer types.

- **Control the rollover of free resources**. For example, you can control rollovers with a rule specifying that when a customer changes plans, free resources can be rolled over only if plans are within the same plan list.

The name and type together identify a unique plan list. The name and type are case sensitive. For example, Gold - Senior and gold - senior are two different plan lists. You can assign any name and type to the plan lists.

You can create any number of plan lists for your database, and each plan list can contain any number of plans. Two different plan lists can contain the same plan.

Copyright © 2025, 2026, Oracle and/or its affiliates.

The plan list does not have to include all of your plans. You can create plans and not include them in a plan list until you need them. Or, you can offer one set of plans to one group of potential customers, and another set of plans to another group.

BRM includes two types of plan lists:

- Use *new* plan lists to register new customers.
- Use *add-on* plan lists to add services to existing accounts.

You can also add your custom plan list based on the customer type.

The plan lists you create are displayed in Customer Center. If you use your own custom application, use the PCM_OP_CUST_POL_GET_PLANS policy opcode to retrieve and display plan lists.

About the Default and CSR Plan Lists

BRM includes special plan lists that you can use to determine where the plans are displayed. For example, you might want some plans to be displayed only in Customer Center.

- Plans in the **CSR - new** and **CSR - addon** plan lists are displayed only in Customer Center.

> **Note**
>
> The **webclient** name is the default but can be changed. See the discussion about specifying which plans to display in *BRM Managing Customers*.

- Plans contained in the **default-new** and **default-addon** plan lists are used in Customer Center if the **CSR** plan list is not available.

About Offer Profiles

Offer profiles are made up of one or more policy labels each of which defines a gradation in the quality of service (QoS) based on usage amounts for a resource.

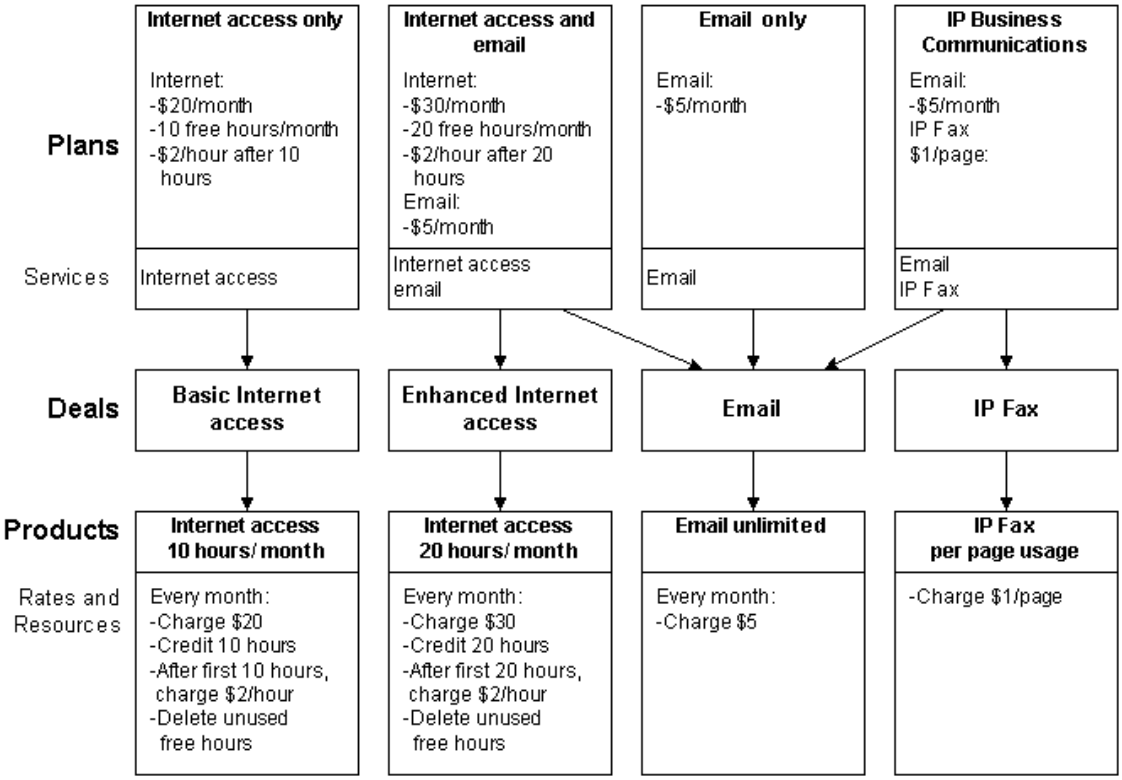
For example, you can have an offer profile called *"Platinum"* for a data service and define its resource as *Megabytes Used*. You can define a policy labeled *Fair Usage*, which has three levels, *Low QoS*, *Medium QoS*, and *High QoS*, with each level containing a usage range valid for that quality of service.

Price List Example

In the sample price list shown in Figure 1-11, three services are offered: Internet access, email, and IP fax service. All rate plans used in this price list are real-time rate plans.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 1-11 Sample Price List



Note the following points about how this price list is organized:

- There are two products that define different monthly fees for Internet access. These products are used by different deals.
- Two of the plans include two deals, the other two plans include only one deal each.

For information about the sample plans, see "Understanding the Sample Pricing Plans".

About Setting Up a Price List

Setting up a price list typically requires the following steps:

1. Marketing and finance personnel define which services your company offers, and how much to charge for them. For example, they define how much to charge each month, how much to charge for online usage, and how much to charge to sign up for a plan.
2. Operations personnel review the pricing model to determine if any new services must be created to support the price list. In addition, they determine if any new resources, services, events, general ledger (G/L) IDs, or tax codes must be created. See "Prerequisites for Creating a Price List".
3. An operations person plans the price list (for example, to determine how many plans and deals to create and to determine the products that make up the plans). This plan is typically reviewed by marketing and finance personnel to confirm that it implements the company's pricing model. See "Planning Your Price List".
4. An operations person uses Pricing Center to create the price list. Because products are the basic unit of the price list, they are created first. See "About Using Pricing Center".

Copyright © 2025, 2026, Oracle and/or its affiliates.

5. An operations person tests the price list. See "Testing Your Price List".

Prerequisites for Creating a Price List

Before you create a price list, you must complete the following tasks:

- **Create services and events**

  BRM includes Internet access and email services by default, but you might need to modify them or add new services before you create your price list. You must also configure a list of events to track for each service. If you create new services, you may need to create new events to track them.

- **Create resources**

  Rate plans require resources. Use the Resource Editor in Pricing Center to add or modify resources. For more information, see "About Resources".

- **Define currency conversion rates**

  Using more than one currency requires that you define currency conversion rates.

- **Create G/L IDs**

  You use G/L IDs to collect general ledger information from the BRM database and export it to your accounting application. You must decide how to track the revenue for each type of rate and create the appropriate G/L IDs.

- **Define tax codes and tax suppliers**

  To calculate taxes, you must define tax codes and tax suppliers.

- **Set up offer profiles**

  To ensure that resource tracking for policy-driven provisioning of services, use the offer profile name as the provisioning tag in the product or discount. Set the resource used in the offer profile as the resource counter in the product or discount.

- **Define product-level provisioning categories**

  If you use product-level provisioning, you must define provisioning tags. See "About Using Product-Level Provisioning to Configure Services" and "Working with Provisioning Tags ".

- **Define ratable usage metrics (RUMs) for events**

  You use RUMs to identify the event attributes to rate for each event. RUM definitions are stored in the BRM database. You use a text editor and a utility to define RUMs. For more information, see "About Ratable Usage Metrics".

- **Map event types to services**

  When you create a product, you select a service and events you want to rate that are related to that service. Because all event types are not valid for all services, you map event types to services. Creating this map prevents you from selecting an event that does not occur for a given service. See "Mapping Events and Services".

- **Define zones**

  For real-time rating, you use zones to create a single value to represent a group of values. You use the representative value in a rate plan selector. See "About Real-Time Zoning".

  For information on zones in batch pipeline rating, see "Setting Up Zones for Batch Pipeline Rating".

- **Define impact categories**

Copyright © 2025, 2026, Oracle and/or its affiliates.

For real-time rating, you use impact categories to specify that a particular group of balance impacts within a rate should be used. If you plan to use attribute value grouping during rating, you must define some impact categories. See "Real-Time Rating Using Event Attributes".

For pipeline rating impact category information, see "About Impact Categories".

- **Define pipeline data**

  If you use pipeline rating, you must define several types of data and pricing components. See "Setting Up Real-Time and Pipeline Pricing and Rating".

Planning Your Price List

When you plan your price list, you determine how much to charge for your services, for example:

- Which types of fees to use for a service, such as a flat monthly fee, hourly usage fees, or both.
- The rates to charge for monthly fees, hourly usage, set up, and so forth.
- Discounts such as those based on time of day, date of purchase, or volume usage.
- How to handle multiple currencies.
- Special pricing options such as sponsored rates.

> **Note**
>
> Because price lists are built by setting up relationships among plans, deals, and products, it is helpful to draw a diagram of your price list before you create it in Pricing Center. For an example, see "Price List Example".

About Using Pricing Center

You use Pricing Center to create and modify your price list.

> **Note**
>
> Offer profiles cannot be created or modified using Pricing Center. Use the **loadpricelist** utility or the PCM_OP_OFFER_PROFILE_SET_OFFER_PROFILE opcode.

With Pricing Center, you can perform the following tasks:

- Create and modify products, deals, plans, and plan lists.
- Create and modify rate plans for products. You can create rate plans for real-time rating and for pipeline batch rating.
- Define complex rate plans for your services based on valid time periods, event quantities, and rate priorities.
- Set credit limits and discounts.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Reuse deals, plans, products, and rate plans so you do not need to re-create them every time.
- Update your pricing model without stopping and restarting your BRM system.
- Save a pricing document to a binary file and open the file at a later date. This enables multiple people to work on different pricing objects that will be committed to the same database.

Who Uses Pricing Center?

Generally, the services you offer and how you charge for them are defined by your marketing and finance personnel. The price list is typically created by operations personnel using Pricing Center. You should outline, in detail, your entire pricing structure before you use Pricing Center to implement it.

Example of Using Pricing Center

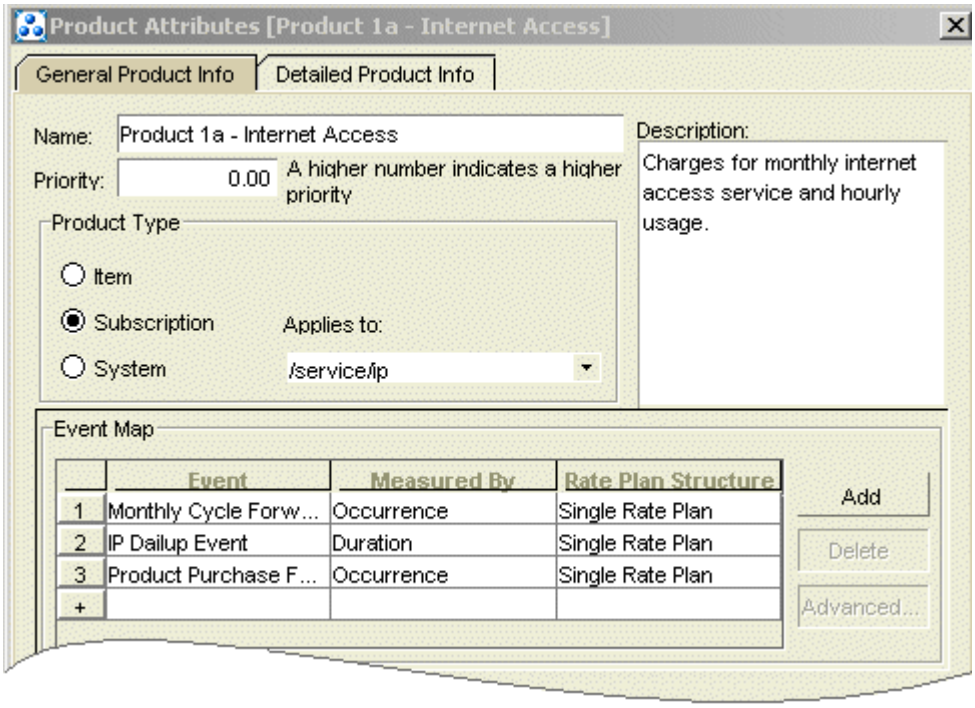
When you are ready to use Pricing Center to create a price list, the price list should be planned, and all necessary components, such as resources, G/L IDs, and RUMs should be in place. (See "Prerequisites for Creating a Price List" and "Planning Your Price List".)

You perform the following steps to create a price list by using Pricing Center:

1. Start by using the Product Creation wizard to create the products. The wizard steps you through the general product parameters, such as the name and description of the product, the service it applies to, and the start and end dates.
2. When you finish, Pricing Center displays the product attributes that you selected. You can change them at any time.
3. After the general product properties are defined, you define the rate plans that specify how much to charge. In the example illustrated in Figure 1-12, the product includes an IP connection fee, a monthly fee, and a purchase fee, so you create rate plans for those events.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 1-12 Product Example



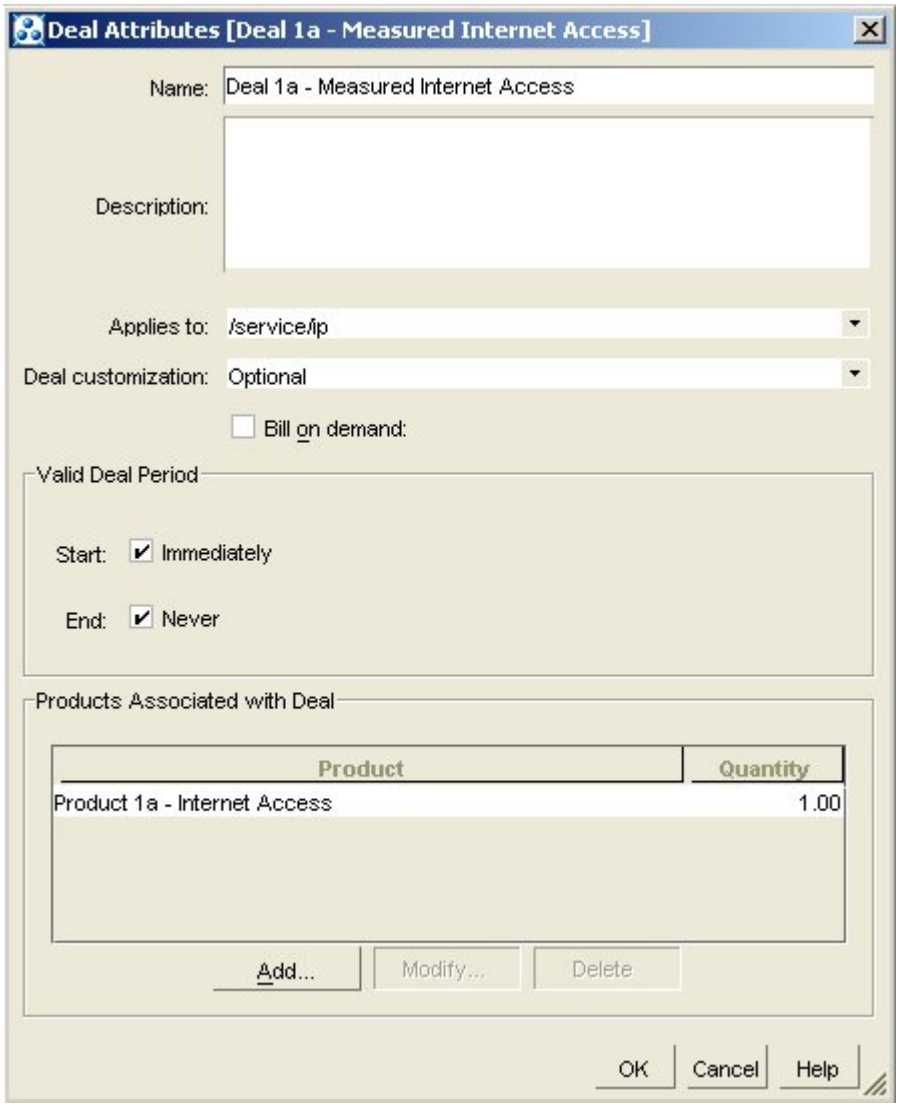
4. For real-time rate plans, you apply balance impacts to each rate plan. Figure 1-13 shows the balance impacts for a purchase fee.

Figure 1-13 Balance Impact Example

5. When the product is finished, you create a deal and add the product as shown in Figure 1-14.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 1-14 Adding Product to Deal Example

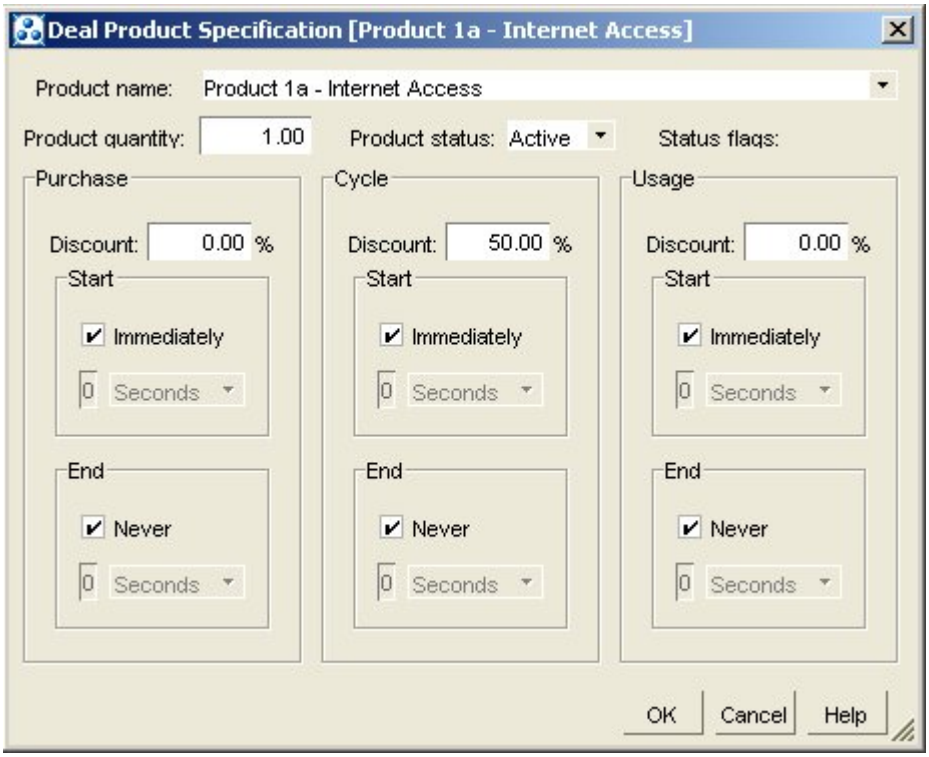


6. Then you tailor the product settings specifically for the deal.

   To discount real-time rates, you can include a discount in the deal. Figure 1-15 shows a deal that provides a 50% discount for the first month cycle fee.

Copyright © 2025, 2026, Oracle and/or its affiliates.

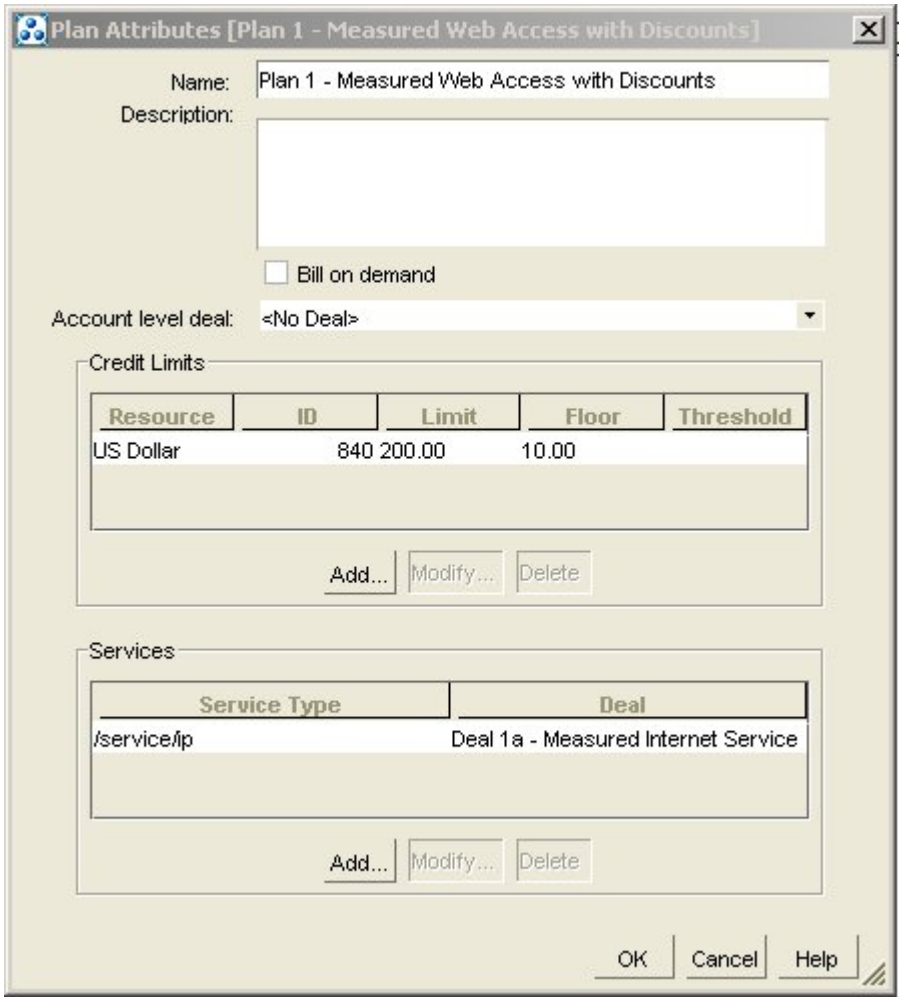
Figure 1-15 Discount Example



7. After defining deal-specific values for the products, you create a plan and add the deal to the plan. In the plan, you set the credit limit. In Figure 1-16, the credit limit is $200.

Copyright © 2025, 2026, Oracle and/or its affiliates.

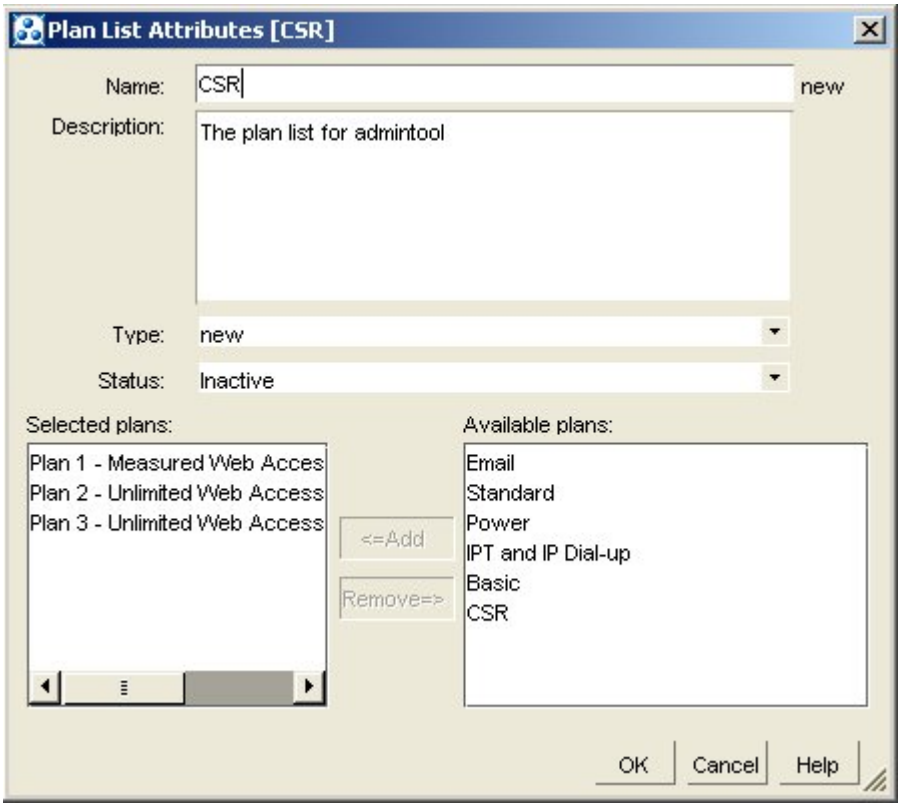
Figure 1-16 Plan Attributes Example



8. After creating the plan, you add it to one or more plan lists. In Figure 1-17, the plan is added to the **CSR - new** plan list. This plan list is displayed in Customer Center.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 1-17 Plan List Attributes Example



9. After adding the plan to the plan list, choose **File - Commit** to commit the price list to the database.

   You test the price list by creating accounts that use your new plan, generating activity, and running billing to ensure that the account balances are impacted correctly. See "Testing Your Price List".

Making Changes to Your Price List

You can make changes to your price list at any time. For example, if you offer a new service, you can create new plans and deals to charge for that service.

You can also change rates for existing products, add products to existing deals, and so forth.

Deleting Products

You cannot delete a product from the price list if it is owned by any account. To delete a product, cancel it in all accounts.

Logging Changes to Price Lists

To get notified when price lists change, BRM can create messages in the **cm.pinlog** file when price lists are updated. For information about **cm.pinlog** and other log files, see the discussion about using logs to monitor components in *BRM System Administrator's Guide*.

To log price list changes, set the **fm_rate log_refresh_product** entry in the Connection Manager (CM) **pin.conf** to **1**. If this entry is absent or set to **0**, BRM does not log changes to the price lists.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> Products which are used while rating events are cached by rating. The cache is refreshed when a product is modified based on the CM **pin.conf** entry **refresh_product_interval**. This entry specifies the time after which the cache needs to be refreshed. If this entry is not specified, the default refresh interval is one hour. When the cache is refreshed and **log_refresh_product** is enabled, a debug message is created in the **cm.pinlog** file indicating the product POID that was refreshed.

To log price list changes:

1. Open the CM configuration file (*BRM_home***/sys/cm/pin.conf**).
2. Set the value for the **fm_rate log_refresh_product** entry to **1**:

   ```
   - fm_rate log_refresh_product   1
   ```

3. Save the file.
4. Stop and restart the CM.

Ensuring Price List Consistency

For implementations covering a large geographic area, you might need regional price lists, each with variations in the pricing structure. In that case, you should maintain consistency in the names of price list elements.

For example, use a common name for noncurrency resources, such as access hours. It is easier to keep track of global usage when a resource has the same name on every regional price list. You should also have common names for usage rates.

Keep in mind that rates are applied independently of the products that include them. If multiple developers create products independently that can be purchased by the same customers, ensure that the rates in those products charge the same amounts and do not have conflicting priorities.

Troubleshooting a Price List

Pricing Center checks the validity of your price list. If you cannot save your price list, check for the following errors:

- **Time conflicts**. For example, you might set the start date for a rate to March 1 and then set the start date for a deal that contains the rate to February 1st. This means that on February 1st, a deal is available for purchase that contains a rate that is not valid for another month.
- **Nonvalid credit limits and credit floors**. A credit limit must be equal to or greater than 0. A credit floor must be less than or equal to the credit limit.

> **Note**
>
> Pricing Center does not include or validate pipeline rating data when you save a price list.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Displaying Price List Elements

You can show a detailed outline of all the elements in a price list, including plans, deals, products, rate plans, and rates by running the **PriceList** report. For more information, see the discussion about price list report in *BRM Reports*.

> **Note**
>
> Offer profiles cannot be displayed using Pricing Center.

Creating a Price List with the XML Pricing Interface

You can use the XML Pricing Interface instead of using Pricing Center to create your price list. See "Using the XML Pricing Interface to Create a Price List ".

Common Price List Solutions

The following topics describe how to implement these common price list solutions:

- Providing Free Hours
- Deleting Unused Free Hours
- Charging for Canceling a Product
- Charging a Discounted Purchase Fee Based on Date of Purchase
- Creating a “Third-Month Free" Rate
- Creating Discounts Based on Usage
- Creating Products for Administrative Events

> **Note**
>
> These topics apply to real-time rating only.

Providing Free Hours

To provide 10 free hours every month, you need three rate plans:

- A cycle forward rate plan that credits 10 free hours every month.
- A usage rate plan with two rate tiers: one that charges hours as the resource until the free hours are gone and one that charges dollars as the resource.
- A fold rate plan that deletes unused hours if the customer does not use all 10 free hours.

For example:

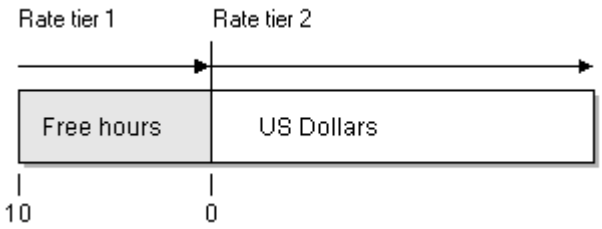
- Cycle forward rate plan: Credit 10 hours per month
- Balance impact: -10 hours every month
- Usage rate plan:

Copyright © 2025, 2026, Oracle and/or its affiliates.

  - Rate tier 1: Charge hours. This tier is valid until the hours in the customer's account reaches 0.
  - Balance impact: 1 hour for every hour online
  - Rate tier 2: Charge dollars. This tier is valid when rate tier 1 can no longer be applied.
  - Balance impact: 1 dollar for every hour online.
- Fold rate plan: Delete unused hours (see "Deleting Unused Free Hours").

Figure 1-18 shows how the free hours are used first.

Figure 1-18 Free Hours Example



Deleting Unused Free Hours

When you credit free hours every month, you probably do not want your customers carrying over unused free hours from one month to the next.

To delete unused free hours, create a cycle fold rate plan that subtracts one hour for every free hour in the customer's balance at the end of the month. After the rate plan removes the left-over free hours, a cycle-forward rate plan applies the monthly credit of free hours.

Figure 1-19 shows the balance impact in Pricing Center. The resource is hours and the balance impact is -1, which removes one hour for every hour in the account balance.

Figure 1-19 Balance Impact that Deletes Unused Hours

	Resource ID	GLID	Impact Category	S	P	D	Fixed Amount	Scaled Amount	Units
1	async bulk hours [100...	undefine...		☐	✔	☐	0.00	-1.00	None [0]

Charging for Canceling a Product

To charge a fee for canceling a product, you include a cancel rate plan in your product.

To create a cancel rate plan that is applied only when closing an account, create a product that has only a cancel rate plan. You can cancel other products without charging a cancel fee, but when the account is closed, the product with the cancel rate plan is canceled, which generates the cancel fee.

Charging a Discounted Purchase Fee Based on Date of Purchase

To charge a discounted purchase fee based on the date of purchase, you need two rate tiers:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- A high-priority rate tier that is valid only in a specified date range. This rate tier charges the discounted purchase fee.
- A low-priority rate tier that is always valid. This rate tier charges the normal purchase fee.

For example:

- Rate tier 1: Discounted purchase fee
- Priority: high
- Balance impact: 7.5 US dollars
- Absolute duration end time: 12:00 a.m. on 12/31/99
- Rate tier 2: Normal purchase fee
- Priority: low
- Balance impact: 15 US dollars

Creating a “Third-Month Free" Rate

To give a customer the third month of a cycle forward fee free, you must create a cycle forward rate plan with two rate tiers:

- Create a high-priority rate tier that uses relative duration to start three months after the product is purchased and has a balance impact of 0.
- Create a low-priority rate tier for the normal cycle forward fee.

For example:

- Rate tier 1: Free month
- Balance impact: 0 US dollars
- Relative duration start time: 55 days
- Relative duration end time: 65 days

  The relative duration start date is 55 days after purchase to ensure that the rate tier is valid before the third billing cycle starts, which, depending on the length of the month, could start within 59 days of the product purchase.

  The relative duration end date is 65 days after purchase to ensure that the rate tier is not valid in the fourth billing cycle.
- Rate tier 2: Normal cycle fee
- Balance impact: 19.95 US dollars

Creating Discounts Based on Usage

To define a rate plan based on volume usage, use multiple quantity discount brackets.

The following example uses two quantity discount brackets; one to charge .02 US dollars for each of the first 100 email messages and another .01 US dollars for each subsequent message.

Which quantity discount bracket is selected depends on the total quantity rated by both brackets.

- Quantity Discount Bracket 1: Balance impacts for first 100 email messages
- Minimum quantity: none

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Maximum quantity: 100
- Balance impact resource: US dollars
- Scaled impact: .02 US dollars
- Quantity Discount Bracket 2: Balance impacts for email messages after the first 100
- Minimum quantity: 100
- Maximum quantity: none
- Balance impact resource: US dollars
- Scaled impact: .01 US dollars

Creating Products for Administrative Events

To create products for administrative events, you must first define usage rates, as explained in "About Charging for Administrative Events".

When you create the products:

- If the event that you're rating is not tied to any service, use the All Accounts/No Services purchase level.
- If the event applies to all customers at all times, use the default product to define the rate.

Advanced Rating Features

For information about advanced rating features, see:

- Charging Cycle Forward Fees in Advance
- About Charging for Administrative Events
- About Using Product-Level Provisioning to Configure Services
- About Custom Event Analysis
- Setting Optional Rating Flags

> **Note**
>
> These topics apply to real-time rating only.

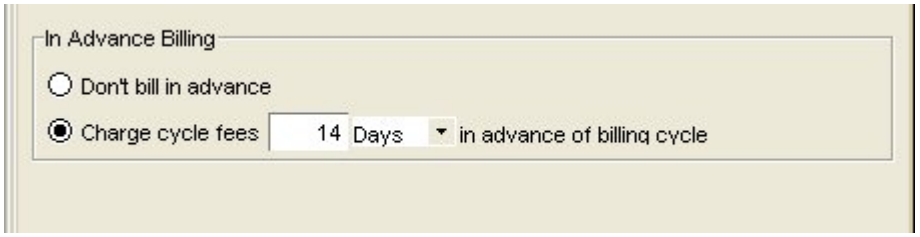
Charging Cycle Forward Fees in Advance

You can set up cycle forward rates to charge cycle forward fees in advance. This enables you to charge all or a portion of the next cycle fee in the current bill.

When you define your event map in Pricing Center, you can specify how far in advance to bill in days, weeks, or months, as shown in Figure 1-20.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 1-20 Advance Billing Example



As part of setting up billing in advance, you must enable proration. Use Pricing Center to enable proration and take the other steps necessary to set up billing in advance.

Example: Charging Cycle Forward Fees in Advance for Monthly Cycle Fees

Suppose billing is monthly and in advance billing is set for 15 days.

When billing occurs on February 1, it charges cycle fee from February 1 to March 1 and additional 15 days to March 15.

When billing occurs on March 1, it charges cycle fee from March 15 to March 31 and additional 15 days to April 15.

When billing occurs on April 1, it charges cycle fee from April 15 to May 15, and so on.

Example: Charging Cycle Forward Fees in Advance for Quarterly Cycle Fees

Suppose billing is monthly and in advance billing is set for one month.

When billing occurs on February 1, it charges cycle fees from February 1 to May 1 and additional one month to June 1.

When billing occurs on March 1 and April 1, no cycle fees are charged.

When billing occurs on May 1, it charges cycle fees from June 1 to August 1 and additional one month to September 1.

When billing occurs on June 1 and July 1, no cycle fees are charged.

When billing occurs on August 1, it charges cycle fees from September 1 to December 1, and so on.

About Charging for Administrative Events

You can charge for *administrative events*, such as changing a password. To do so requires programming. See "About Charging for Custom Events and Attributes".

About Using Product-Level Provisioning to Configure Services

You can use product-level provisioning to define how a service is configured and to rate each configuration differently.

For example, if you have a product that provides an IP service, you can define two service configurations based on the bandwidth of the IP connection. You would create two products, one for each service configuration (for example, Fast Fax and Regular Fax).

Copyright © 2025, 2026, Oracle and/or its affiliates.

To create products that configure services, you assign provisioning tags to the products. For example, to create a Fast Fax product, you would create a Fast_Fax provisioning tag that sets a higher bandwidth for the customer's connection. When a customer purchases the product that includes the Fast_Fax provisioning tag, the fax service is provisioned with the higher bandwidth.

When creating products in Pricing Center, you choose the provisioning tag that identifies the service configuration.

The advantage to configuring services by using products is that you do not need to define different services to handle different service configurations.

You can also use provisioning tags to create extended rating attributes, which enable you to offer special rates and promotions.

To create provisioning tags, see "Working with Provisioning Tags ".

About Remittance

Use the remittance feature to share the revenue you receive with third parties. You can direct BRM to calculate the amount of remittance in various ways. For example, you can pay a percentage of subscriber fees or a flat amount per new subscriber to third parties such as resellers or service providers.

You must create remittance products to use the remittance feature. See "Remitting Funds to Third Parties" in *BRM Configuring and Running Billing*.

About Custom Event Analysis

Custom event analysis enables you to perform rating based on:

- Custom attributes of events or non-event attributes, such as an account attribute or a profile attribute.
- Guidelines that are more complex than the defaults, such as rating based on percentage values or expressions containing a greater than or less than operator.

Setting up custom event analysis requires modifying or creating a policy opcode and editing and loading several configuration files.

Setting Optional Rating Flags

You can turn a number of optional rating features on and off by setting business parameters:

- Changing the way fixed discount amounts are calculated. See "Providing Deal-Level Discounts".
- Turning credit floor checking on and off. See "Credit Limit and Floor Options".
- Determining whether to apply a credit limit when an account does not include a balance for a particular resource. See "Credit Limit and Floor Options".
- Determining whether folds for canceled (inactive) products should be rated.
- Determining whether the PIN_FLD_EXTENDED_INFO substruct field should be returned with the rating results. This field can be used to transport reusable information among transactions. For more information, see the input flist specification for PCM_OP_RATE_EVENT.

To use these features:

Copyright © 2025, 2026, Oracle and/or its affiliates.

1. Go to *BRM_home*/**sys/data/config**.
2. Use the following command to create an editable XML file from the **Rating** instance of the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsRating bus_params_rating.xml
   ```

   This command creates an XML file named **bus_params_rating.xml.out** in your current directory. If you do not want this file in your current directory, specify the path as part of the file name.
3. In **bus_params_rating.xml.out**, set the following entry:

   ```
   <ExtraRateFlags>value</ExtraRateFlags>
   ```

   where *value* is one of the following:

   - **0x01**: Specifies to use a fixed discount.
     If present, discounts are calculated using this formula:

     [(rate – discount) * quantity]

     If absent, discounts are calculated using this formula:

     [(rate* quantity) – discount]
   - **0x02**: Specifies to disable credit floor checking.
   - **0x10**: Specifies to return extended information with the rating result.

   > ⚠ **Caution**
   >
   > BRM uses the XML in this file to overwrite the existing instance of the **/config/business_params** object. Use care when updating parameters in the file.
4. Save and exit the file.
5. Rename the **bus_params_rating.xml.out** file to **bus_params_rating.xml**.
6. Use the following command to load your changes into the **/config/business_params** object:

   ```
   pin_bus_params bus_params_rating.xml
   ```

   You should run this command from the *BRM_home*/**sys/data/config** directory, which includes support files used by the utility. To run it from a different directory, see "pin_bus_params" in *BRM Developer's Guide*.
7. Read the object with the **testnap** utility or the Object Browser to verify that all fields are correct.
   For general instructions on using **testnap**, see "Using the testnap Utility to Test BRM" in *BRM Developer's Guide*. For information on how to use Object Browser, see "Reading Objects" in *BRM Developer's Guide*.
8. Stop and restart the Connection Manager (CM).
   For more information, see "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.

Copyright © 2025, 2026, Oracle and/or its affiliates.

2

Understanding the Sample Pricing Plans

This chapter describes the sample pricing plans included with Oracle Communications Billing and Revenue Management (BRM) Pricing Center.

Topics in this document:

- About the Sample Plans
- Descriptions of the Sample Plans
- Re-Creating the Sample Plans
- Descriptions of the Advanced Sample Plans

For information about creating a price list, see "About Creating a Price List".

About the Sample Plans

The following sections describe the sample price plans.

Purpose of the Sample Plans

The sample BRM pricing plans described in this chapter show a range of possibilities for creating typical plans with BRM. Also included are some advanced sample plans to illustrate pricing and rating strategies. You can use the samples as templates for creating your own pricing plans. You might find a plan that matches exactly the plan you want to offer to your customers. More likely, you will see elements of several plans that you want to include in your own plan. You can pick and choose which elements you want and then see how those elements were created, making it easy to incorporate them into your plan.

Looking at the Sample Plans

BRM provides price list documents (files with the IPL or XML extension) that you can display in the Pricing Center:

> **Note**
>
> Offer profile data cannot accessed using Pricing Center. Use the **loadpricelist** utility or the PCM_OP_OFFER_PROFILE_SET_OFFER_PROFILE opcode.

- Basic BRM plans for Internet service providers (**basicISP.ipl**) are located by default in the **Sample_Price_Plans** subdirectory of the Pricing Center installation directory.
- Plans that are based on a BRM optional manager are located by default in the **Optional_Manager_Plans** subdirectory of the Pricing Center installation directory.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> If you're implementing an optional manager, it is a good idea to familiarize yourself with the basic plans before creating a price list for an optional manager.

- The advanced sample plans are located by default in the **Sample_Price_Plans** subdirectory of the Pricing Center installation directory.

Opening an IPL Price List File

1. Start Pricing Center.
2. Choose **File - Open**.
3. Open the **PricingCenter\Sample_Price_Plans** directory.
4. Select the IPL file and click **Open**.

Opening an XML Price List File

1. Start Pricing Center.
2. Choose **File - Import > Real-time Data**.
3. Select the XML file and click **Open**.

   You must connect to a database to access configuration information, such as resources, G/L IDs, and valid events for services.
4. Enter the database information in the Welcome dialog box.

> **Note**
>
> Click the **Help** button for information on login values.

Descriptions of the Sample Plans

This section describes each of the sample plans in the language your company's marketing department might present to your customers. Each plan includes a brief description, the target customer, and the key elements of the plan.

For step-by-step procedures on creating these plans, see "Re-Creating the Sample Plans".

Plans for Internet Service Providers

This section describes the sample plans for internet service providers.

Plan 1 – Measured Web Access with Discounts

This plan provides Internet access for a basic monthly fee, $9.95 in this example, plus an hourly fee that is discounted for usage outside of prime time.

Target customer

New Web user who might adjust access times to take advantage of discount periods

Copyright © 2025, 2026, Oracle and/or its affiliates.

Concepts illustrated

- Simple monthly (cycle forward) fee for Internet access
- Prorating of cycle fees
- Measured rates
- Time-of-day usage rates
- Priority assignments for multiple rates
- Product purchase limited to a set maximum

Key elements of plan

- Internet access service
- $9.95 basic monthly rate
- Measured rate of $2.00/hour for prime-time access (M-F, 8 a.m. to 6 p.m.)
- Measured rate of $1.00/hour for access outside of prime time
- First month prorated
- No partial refund for cancellation in mid-cycle
- Email service
- A single email address available for an additional $3.00/month

To re-create this plan

See "Re-Creating Plan 1 – Measured Web Access with Discounts".

Plan 2 – Unlimited Web Access with Discounts

This plan provides unlimited Internet access and one email address for a fixed monthly fee, $19.95 in this example. There's an initial sign-up fee, discounted when the plan is purchased before a certain date. The third month is free.

Target customer

Heavy-duty Internet user who might be encouraged to stay with the plan by the offer of discounts

Concepts illustrated

- Monthly (cycle forward) fees for Internet access
- Purchase fee
- Discount period on purchase
- Single free period

Key elements of plan

- Internet access service
- Sign-up fee of $15.00
- Sign-up fee discounted by 50% through May, 2000
- $19.95 monthly rate, billed on the first day of the month

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Third month free
- One email address included

To re-create this plan

See "Re-Creating Plan 2 – Unlimited Web Access with Discounts".

Plan 3 – Unlimited Internet Service with Recurring Discounts

This plan provides unlimited Internet access and one email address for a fixed quarterly fee of $45, payable at the beginning of each quarterly period. For every year with this plan, there's a bonus of one free month.

Target customer

Frequent Internet user who might pay in advance for a lower basic rate and who might be encouraged to stay with the plan by the offer of recurring discounts

Concepts illustrated

- Multi-month (cycle forward) fee
- Measured fees
- Recurring free period
- Custom BRM resources
- Reusing products from other plans

Key elements of plan

- Internet access service
- $45.00 per quarter, billed on the first day of every third month
- Last month of each year is free
- One email address included

To re-create this plan

See "Re-Creating Plan 3 – Unlimited Internet Service with Recurring Discounts".

Re-Creating the Sample Plans

This section shows how each of the sample pricing plans was created by using Pricing Center. For more information on specific procedures, see Pricing Center Help.

Overview of Creating a Pricing Plan

Your company's marketing department writes a proposal for offering services to your customers. You create a plan that translates that proposal into the language that BRM uses to keep track of the rates and then bill for customer usage.

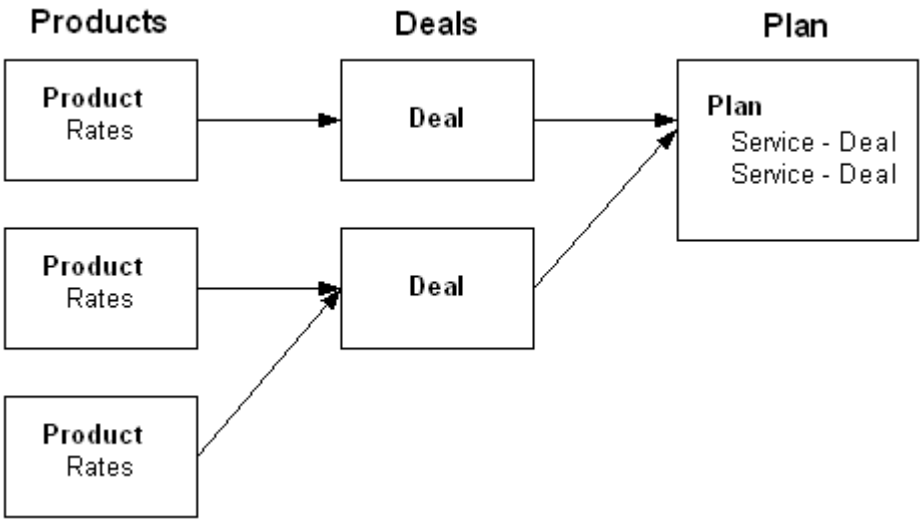
The *pricing plan* shows one or more services that you offer your customers and the terms under which you provide those services. The terms and conditions of a pricing plan is called a *deal*. Each deal consists of one or more *products*, which show the rates charged for a service. You use products to map rates to events, so that when the event occurs in the BRM system, the rate is charged.

Copyright © 2025, 2026, Oracle and/or its affiliates.

To create a plan, you define the products and then add them to a deal, which you bundle into a plan and associate with your service. In some cases you might also need to create resources or impact categories for a plan.

Figure 2-1 shows the building blocks of a pricing plan.

Figure 2-1 Pricing Plan Building Blocks



To create a plan, follow these general steps:

1. Define the key elements of the plan: what services you want to offer and what you want to charge for them.

   The descriptions of the sample plans show the key elements of these plans.
2. Use Pricing Center to create a product for each key element of your plan. Usually, each service you offer uses one product to contain all rates for that service.
3. Use Pricing Center to group the products into one or more deals.
4. Use Pricing Center to create a pricing plan that associates the deals with the services.

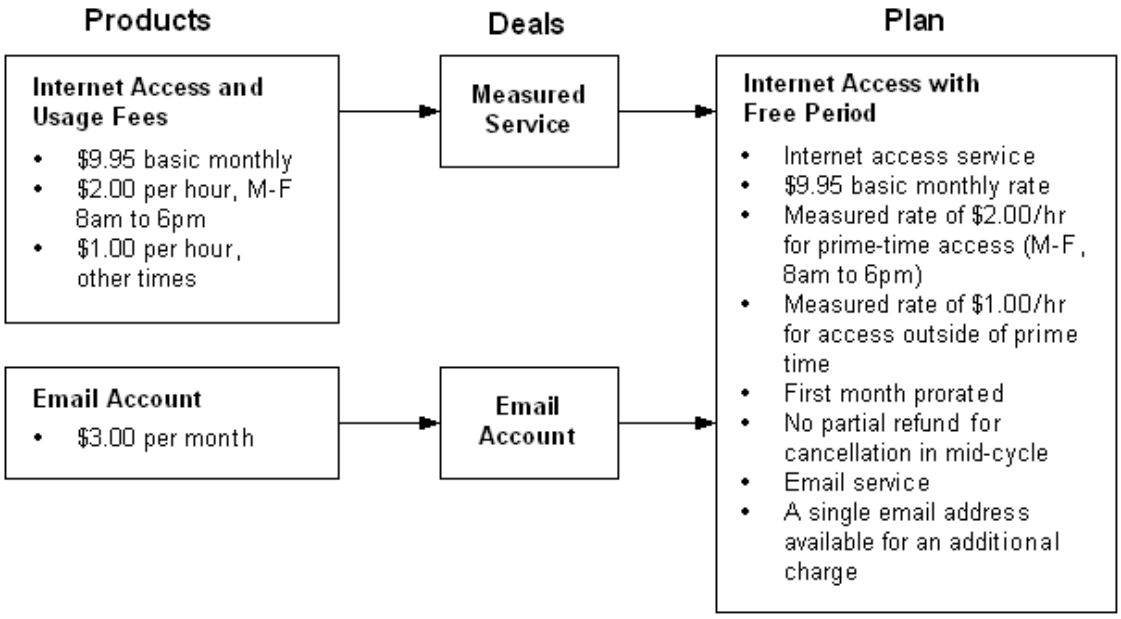
Plans for Internet Service Providers

Figure 2-2 show the Measured Web Access with Discounts Plan used in the following example.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Re-Creating Plan 1 – Measured Web Access with Discounts

Figure 2-2 Measured Web Access with Discounts Plan



Putting together Plan 1

1. Create a product to charge for the Internet access service and Internet measured usage. See "Product 1a – Internet Access".
2. Create a product to charge for the email account. See "Product 1b – Email Account".
3. Create a deal for the Internet access. See "Deal 1a – Measured Internet Service".
4. Create a deal for the email account. See "Deal 1b – Standard Email Access".
5. In Pricing Center, choose **Insert – Plan**.
6. Name the plan: **Plan 1 – Measured Web Access with Discounts**. (You can also add a description.)
7. In the **Services** section, click **Add**.
8. For **Service Type**, select **/service/ip**.
9. For **Deal**, select **Deal 1a – Measured Internet Service**.
10. Click **OK**.
11. In the **Services** section, click **Add**.
12. For **Service Type**, select **/service/email**.
13. For **Deal**, select **Deal 1b – Standard Email Access**.
14. Click **OK**.
15. In the **Credit Limits** section, click **Add**.
16. In the **Resource ID** list, select **US Dollar**.

Copyright © 2025, 2026, Oracle and/or its affiliates.

17. In the **Credit Limit** box, type **250**.
18. Double-click **OK** to finish the plan.

Product 1a – Internet Access

The purpose of this product is to charge for the Internet access service at the beginning of each month and for time spent connected to the Internet when it occurs. The product has three rates:

- One for the monthly fee
- One for usage during prime-time hours ($2.00/hour)
- One for usage during off-peak hours ($1.00/hour)

You create the fees by mapping the rates to *events*, which define how the charges occur.

Product 1a - Defining the Product

Table 2-1 describes the steps for defining the product.

Table 2-1 Defining the Product

Do This	Why
In Pricing Center, choose **Insert – Product** and select **Create a new product**.	To create a product.
Click **Next** and give the product a name: **Product 1a – Internet Access**. (You can also add a description.) Click **Next**.	The product name identifies the product in Pricing Center. This name must be unique to all products in the same database brand.
When asked what type of product to create, select **Subscription** and in **Applies To**, choose **/service/ip**.	This enables the product to contain recurring monthly and usage rates and associates it with the IP service.
Finish the product creation wizard, and click **Yes** at the end. The Product Attributes window opens.	This product uses the default values so you do not need to specifically set them.

Product 1a - Creating the Monthly Internet Fee

You must create an event rating map to charge the customer an access fee each month for the IP service. The access fee is mapped to the monthly cycle event so fees can be tracked and posted to the customer's account when the cycle event occurs.

Table 2-2 describes the steps for creating the Monthly Internet Fee.

Table 2-2 Creating the Monthly Internet Fee

Do This	Why
In the **Event Map** area of the Product Attributes window, click **Add**. In the **Event** column, select **Monthly Cycle Forward Event**.	To create the monthly access fee by associating the rate details with the cycle event, as specified by the plan. When you select **Monthly Cycle Forward Event**, Pricing Center automatically selects **Occurrence** in the **Measured By** column and **Single Rate Plan** in the **Rate Plan Structure** column. Leave these values as they are.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 2-2 (Cont.) Creating the Monthly Internet Fee

Do This	Why
Click **Open Rate Plan**.	To set up the rates for this product.
In the **Plan Detail** tab, enter **cycle_forward_monthly** in the **Name** field.	The rate plan name identifies the rate plan in the product and reminds you of its purpose.
In the **Rate Structure** tree view, select **Tier 1**. In the **Tier Detail** tab, enter a tier name: **Monthly Service Fee**.	The name identifies the rate tier in the product and reminds you when it is valid.
In the **Tier Detail** tab, select either **Absolute** or **Relative**, then select **Starts Immediately** and **Never Ends**. **Note:** When you select **Starts Immediately** and **Never Ends**, it does not matter whether you select **Absolute** or **Relative** effective dates.	To make the rate available indefinitely, so it never expires in the BRM database.
In the **Rate Structure** tree view, select **Rate 1**. In the **Rate Detail** tab, enter a rate name: **$9.95 a month**. **Note:** Rate names must be unique. In the **/rate** storable class, the rate name is specified in the PIN_FLD_DESCR field.	To remind you of the purpose of this rate.
In the **Rate Data** tab, select **Overrides credit limit**.	To continue to charge for Internet access when customers exceed the credit limit.
In the **Rate Data** tab, select **Continuous** in the **Based on** list.	This option is the default when you use only one group of balance impacts to rate an event.
Click the **Proration** tab.	To specify how to prorate the charges.
In the **Purchase Proration Information** group, click **Calculate the charge based on the amount used**.	To charge customers only for the portion of the first month when they owned the product. **Note:** Most companies base the monthly accounting cycle on the day the customer purchases the product rather than on a calendar month, so proration is not necessary.
In the **Cancel Proration Information** group, click **Charge for the entire cycle**.	To specify that the customer is charged the full monthly fee, even if the customer cancels the service in the middle of the accounting cycle.
In the **Rate Structure** tree view, select the item at the bottom of the rate plan tree to show the **Balance Impacts** tab.	To define the cost of the cycle rate.
Under **Valid Quantities**, select **None** for both the **Minimum** and **Maximum** fields.	The monthly fee is not based on a specific amount of the event.
Click **Add**. **Tip:** Drag the right edge of any column heading to make the list wider.	To add a row to the impacts list and to specify the balance impacts **-** what you charge the customer for the service.
Click **Resource ID** and scroll to select **US Dollar [840]**.	To bill in US dollars.
Click the **GLID** and scroll to select **Monthly Fees [102]**.	To record the charges as monthly fees in your general ledger.
Click the **S** check box to select **Sponsorable**.	To enable the rate to be paid by another account. Select this option to enable one account to pay for another.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 2-2 (Cont.) Creating the Monthly Internet Fee

Do This	Why
Click the **P** check box to select **Proratable**.	Although you already specified the terms of proration, you now confirm that this rate can be prorated.
Click the **D** check box to select **Discountable**.	To enable the rate to be discounted when the service is purchased.
In the **Scaled Amount** column, type **9.95**.	Recurring charges, such as the monthly fee in this product, are called *scaled* rates. Enter the dollar amount of the charge, which is $9.95 per month.
In **Units**, select **None [0]**.	This rate uses the **Occurrence** ratable usage metric (RUM), which is not measured in units.
Click **OK**.	You are finished creating the monthly fee and now must create the usage fees.

Product 1a - Creating the Measured Usage Fees

You must create an event rating map to charge the customer for time spent connected to the Internet. You map the hourly rates to an IP session event so fees can be tracked and posted to the customer's account when the customer uses the service. Although the rates are mapped to the same event, you differentiate between the two by specifying the available time for the rates: one for usage during prime-time hours ($2.00/hour) and one for usage during off-peak hours ($1.00/hour).

Table 2-3 describes the steps for creating the Measured Usage Fees.

Table 2-3 Creating the Measured Usage Fees

Do This	Why
In the **Event** group, click **Add**. In the **Event** column, select **Session Event**.	To associate rate details with the IP session event, so fees can be tracked and posted to the correct account when customers connect to the Internet, as specified by the plan. When you select **Session Event**, by default Pricing Center selects **Duration** in the **Measured By** column and **Single Rate Plan** in the **Rate Plan Structure** column. Do not change these values.
Click **Advanced**.	To display the Event Attributes dialog box.
In **Time of day mode**, select **Timed**.	To specify that the fees depend on the length of the session.
In **Time zone mode**, select **Server**.	To specify that this event will be rated by using the time zone in which the BRM server is located.
In **Minimum event quantity**, enter **1** and select **Minutes** as the unit.	To set the minimum session time, even if the session lasts less than a minute.
In **Rounding Increment**, enter **1** in **Units** and select **Minutes** as the unit. In **Rounding rule**, select **Up**.	To round the session length up to the next whole minute, so seconds are not recorded. For example, if a session lasts 1 minute and 10 seconds, the session is recorded as 2 minutes.
Click **OK**.	To return to the **Product Attributes** dialog box.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 2-3 (Cont.) Creating the Measured Usage Fees

Do This	Why
In the **Single Rate Plan** group, click **Open Rate Plan**.	To set up the rates for this product.
In the **Plan Detail** tab, enter **Internet Usage Fee** in the **Name** field.	The rate plan name identifies the rate plan in the product and reminds you of its purpose.
In the **Rate Structure** tree view, select **Tier 1**. In the **Tier Detail** tab, enter a name for the rate tier: **Hourly Fee for Basic Internet Service.**	The name identifies the rate tier in the product and reminds you when it is valid.
In the **Tier Detail** tab under **Effective Dates**, select **Absolute**.	To enable the time-of-day settings for the rate.
In the **Use Time/Day Restrictions** group, select **Time restrictions**. (In the **Valid Days** tab, leave the default settings.)	To add a time tier to the rate.
In the **Rate Structure** tree view, select **New Time of Day Range**. At the top of the **Valid Times** tab, replace the **New Time of Day Range** text with a name for the hourly range: **8am-6pm**.	The name identifies the level in the rate tier and reminds you when it is valid.
Leave the start time as **08:00 AM**, click the **08:00 PM** end time control and use the down arrow to set **06:00 PM (18:00 hours)**.	This sets the rate as valid from 8 a.m. to 6 p.m.
Under **Rate Structure,** select **Rate 1. In the Rate Data tab, enter** a name for the rate: **$2 per hour**.	The name identifies the amount of the fee in the 8 a.m. to 6 p.m. rate tier.
Select **Overrides credit limit**.	To continue to charge for Internet access if customers exceed their credit limit.
In the **Based on** list, select **Continuous**.	This option is used to rate time-of-day events because the amount charged by the first rate tier can affect the which balance impacts are used in the second rate tier.
In the **Rate Structure tree view,** select **No Minimum - No Maximum** to display the **Balance Impacts** tab.	This displays the **Balance Impacts** tab, where you define the cost of the rate.
Under **Valid Quantities**, select **None** for both the **Minimum** and **Maximum** fields.	The monthly fee is not based on a specific amount of the event.
Under **Balance Impacts**, click **Add**. **Tip:** Drag the right edge of any column heading to make the list wider.	To add a row to the list of balance impacts and to specify the balance impacts **-** what you charge the customer for the service.
In the **Resource ID** list, select **US Dollar [840]**.	To bill in US dollars.
In the **GLID** list, select **Dialup Usage Fee [104]**.	To record the charges as Internet usage fees in your general ledger.
Click **S for Sponsorable**.	To enable the rate to be paid by another account. Select this option to enable one account to pay for another.
In the **Scaled** box, type **2**.	Hourly fees are considered *scaled* rates. You enter the dollar amount of the charge per hour.
In **Units**, select **Hour [3]**.	This is the increment you use to measure and charge for the usage. You are finished creating this rate, which applies between 8:00 a.m. and 6:00 p.m.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 2-3 (Cont.) Creating the Measured Usage Fees

Do This	Why
In the **Rate Structure** tree view, select the **Hourly Fee for Basic Internet Service** tier and click **Add**.	To create the second (off-peak) usage fees.
At the top of the **Valid Times** tab, replace the **New Time of Day Range** text with a name for the hourly range: **6pm-8am**. Press **Enter**.	The name identifies the valid time of the rate.
Click the **08:00 AM** start time control and set it to **06:00 PM (18:00 hours)**. Then set the end time control to **08:00 AM**.	This sets the rate as valid from 6 p.m. to 8 a.m.
In the **Rate Structure** tree view under **6pm - 8am**, select **New Rate**.	This displays the Rate Data tab, where you define the fees for off-peak hours.
In the Rate Data tab, enter a rate name: **$1 per hour**.	The name identifies amount of the 6 p.m. - 8 a.m. rate tier.
Select **Overrides credit limit**.	To continue to bill for Internet access if customers exceed the credit limit.
In the **Based on** list, select **Continuous**.	This option is used to rate time of day events because the amount charged by the first rate tier can affect the discount bracket used in the second rate tier.
In the **Rate Structure** tree view, under the rate name **$1 per hour**, select **No Maximum - No Minimum**.	To display the **Balance Impacts** tab.
Under **Valid Quantities**, select **None** for both the **Minimum** and **Maximum** fields.	There is no minimum or maximum beyond which this rate would not be valid.
Under **Balance Impacts**, click **Add**. **Tip:** Drag the right edge of any column heading to make the list wider.	To add a row to the list of impacts and to specify the balance impacts **-** what you charge the customer for the service.
In the **Resource ID** list, select **US Dollar [840]**.	To charge in US dollars.
In the **GLID** list, select **Dialup Usage Fee [104]**.	To record the fees as dialup usage fees in your general ledger.
Click **S for Sponsorable**.	To enable the rate to be paid by another account. Select this option to enable one account to pay for another.
In the **Scaled Amounts** field, type **1**.	Hourly charges are considered *scaled* rates. You enter the dollar amount of the charge per hour. You are finished specifying the rate for the weekday range. Now you must create the off-peak rate that is valid on the weekends.
Click **OK** until you return to the Document window.	You are finished specifying the usage fees for this product.

Product 1b – Email Account

The purpose of this product is to charge for the single email account available with this plan.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Product 1b - Defining the Product

Table 2-4 describes the steps for defining the product.

Table 2-4 Defining the Product

Do This	Why
In Pricing Center, choose **Insert – Product** and select **Create a new product**.	To create a product.
Click **Next** and give the product a name: **Product 1b – Email Account**. (You can also add a description.) Click **Next**.	The product name identifies the product in Pricing Center. This name must be unique to all products in the same database brand.
When asked what type of product to create, select **Subscription** and in **Applies To**, choose **/service/email**.	This enables the product to contain recurring monthly rates and associates it with the email service.
Continue through the product creation wizard until you get to the **Valid Purchase Quantity** group.	Although you can change various attributes of the product while using the wizard, this sample product uses the default attributes up to this point.
Click **Maximum of** and enter **1**.	To specify that a customer can purchase no more than one email account.
Finish the product creation wizard, and click **Yes** at the end.	After you finish the wizard, you specify the rates for this product.

Product 1b – Creating the Monthly Email Fee

You must create an event rating map to charge the customer a fee each month for the email service. The email fee is mapped to the monthly cycle event so fees can be tracked and posted to the customer's account when the cycle event occurs.

Table 2-5 describes the steps for creating the Monthly Email Fee.

Table 2-5 Creating the Monthly Email Fee

Do This	Why
In the **Event Map** group, click **Add**. In the **Event Type** column, select **Monthly Cycle Forward Event**.	To create the monthly access fee by associating the rate details with the cycle event, as specified by the plan. When you select **Monthly Cycle Forward Event**, by default Pricing Center selects **Occurrence** in the **Measured By** column. You cannot change this value. **Note:** The values in the **Rating Details** group are only used for time-dependent fees.
In the **Rate Plan Structure** column, select **Single Rate Plan**. Under **Single Rate Plan**, click **Open Rate Plan**.	To set up the rates for this product.
In the **Plan Detail** tab, enter **cycle_forward_monthly** in the **Name** field.	The rate plan name identifies the rate plan in the product and reminds you of its purpose.
In the **Rate Structure** tree view, select **Tier 1**. In the **Tier Detail** tab, enter a name for the rate tier: **Monthly Email Fee**.	The name identifies the rate tier in the product and reminds you when its valid.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 2-5 (Cont.) Creating the Monthly Email Fee

Do This	Why
Under **Tier Detail**, select **Absolute** or **Relative**, and then **Starts Immediately** and **Never Ends**. Also, under **Use Time/Day Restrictions**, select **No restrictions**.	To make the rate available indefinitely, so it never expires in the BRM database and the fee is always charged.
In the **Rate Structure tree view,** select **Rate 1. In the Rate Detail tab, enter** a name for the rate: **$3 a month**. **Note:** Rate names must be unique. In the **/rate** storable class, the rate name is specified in the PIN_FLD_DESCR field.	To remind you of the purpose of this rate.
Select **Overrides credit limit**.	To continue to charge for the email account when customers exceed the credit limit.
In the **Based on** list, select **Continuous**.	This option is the default when you use only one quantity discount bracket to rate an event.
Click the **Proration** tab.	To specify how to prorate the charges.
In the **Purchase Proration Information** group, click **Calculate the charge based on the amount used**.	To charge customers only for the portion of the first month when they owned the product. **Note:** Most companies base the monthly accounting cycle on the day the customer purchases the product rather than on a calendar month, so proration is not necessary.
In the **Cancel Proration Information** group, click **Charge for the entire cycle**.	To specify that the customer is charged the full monthly fee, even if the customer cancels the service in the middle of the accounting cycle.
In the **Rate Structure tree view under $3 a month**, select **No Minimum - No Maximum** to display the **Balance Impacts** tab.	This displays the **Balance Impacts** tab, where you define the fee.
Under **Valid Quantities**, select **None** for both the **Minimum** and **Maximum** fields.	The monthly fee is not based on a specific amount of the event.
Under **Balance Impacts**, click **Add**. **Tip:** Drag the right edge of any column heading to make the list wider.	To add a row to the impacts list and to specify the balance impacts **-** what you charge the customer for the service.
Click the **Resource ID** box and scroll to select **US Dollar [840]**.	To charge in US dollars.
In the **GLID** list, select **Monthly Fees [102]**.	To record the charges as monthly fees in your general ledger.
Click **S for Sponsorable**.	To enable the rate to be paid by another account. Select this option to enable one account to pay for another.
In the **Scaled** box, type **3**.	Recurring fees, such as the monthly fee in this product, are called *scaled* rates. You enter the dollar amount of the fee.
In **Units**, select **None [0]**.	This rate uses the **Occurrence** ratable usage metric (RUM), which is not measured in units.
Click **OK** until you return to the Document window.	You are finished creating the email product.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Deal 1a – Measured Internet Service

1. In Pricing Center, choose **Insert – Deal**.
2. Name the deal: **Deal 1a – Measured Internet Service**. (You can also add a description.)
3. In **Applies to**, select **/service/ip**.
4. In the **Products Associated with Deal** group, click **Add**.
5. In the **Product Name** list, select **Product 1a – Internet Access**.
6. Click **OK**.

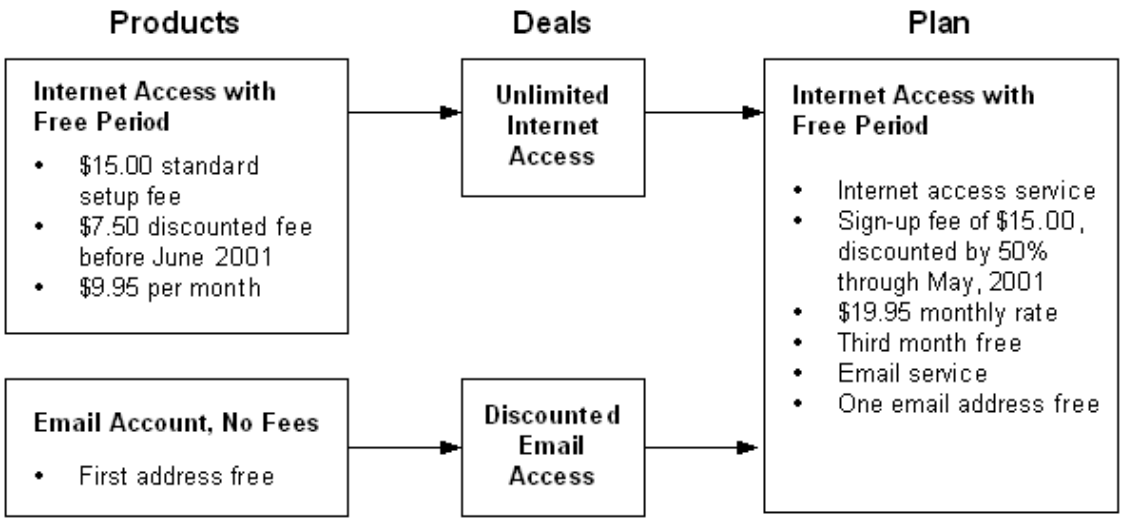
Deal 1b – Standard Email Access

1. In Pricing Center, choose **Insert – Deal**.
2. Name the deal: **Deal 1b – Standard Email Access**. (You can also add a description.)
3. In **Applies to**, select **/service/email**.
4. In the **Products Associated with Deal** group, click **Add**.
5. In the **Product Name** list, select **Product 1b – Email Account**.
6. Click **OK**.

Re-Creating Plan 2 – Unlimited Web Access with Discounts

Figure 2-3 show the Unlimited Web Access with Discounts Plan used in the following example.

Figure 2-3 Unlimited Web Access with Discounts Plan



Putting Together Plan 2

1. Create a product to charge the sign-up fee and monthly Internet access fee. See "Product 2a – Internet Access with Free Period".
2. Create a product for the email account. See "Product 2b – Email Account, No Fees".

Copyright © 2025, 2026, Oracle and/or its affiliates.

3. Create a deal for the Internet access. See "Deal 2a – Unlimited Internet Service".
4. Create a deal for the email account. See "Deal 2b – Discounted Email Access".
5. In Pricing Center, choose **Insert – Plan**.
6. Name the plan: **Plan 2 – Unlimited Web Access with Discounts**. (You can also add a description.)
7. In the **Services** section, click **Add**.
8. For **Service Type**, select **/service/ip**.
9. For Deal, select **Deal 2a – Unlimited Internet Service**.
10. Click **OK**.
11. In the **Services** section, click **Add**.
12. For **Service Type**, select **/service/email**.
13. For Deal, select **Deal 2b – Discounted Email Access**.
14. Click **OK** to finish the plan.

Product 2a – Internet Access with Free Period

The purpose of this product is to charge the customer at the time of registration, and then monthly for Internet access.

Product 2a – Defining the Product

Table 2-6 describes the steps for defining the product.

Table 2-6 Defining the Product

Do This	Why
In Pricing Center, choose **Insert – Product** and select **Create a new product**.	To create a product.
Click **Next** and give the product a name: **Product 2a – Internet Access with Free Period**. (You can also add a description.)	The product name identifies the product in Pricing Center. This name must be unique to all products in the same database brand.
When asked what type of product to create, select **Subscription** and in **Applies To**, choose **/service/ip**.	This enables the product to contain recurring monthly rates and associates it with the IP service.
Continue through the rest of the product creation wizard, and click **Yes** at the end.	Although you can change other attributes of the product while using the wizard, this sample product uses the remaining default attributes.

Product 2a – Creating the Sign-up Fees

You must create an event rating map so the sign-up fees are charged when the purchase event occurs. You use two rate tiers to define the fees. The first rate tier is for a discounted sign-up fee of $7.50, and the second rate tier is for a standard sign-up fee of $15.

Product 2a – Creating the Discounted Sign-up Fee

You want BRM to use the $7.50 discounted rate until it expires. Creating it first in the rate plan gives it a higher priority and ensures that BRM uses this rate during the valid discount period, instead of defaulting to the standard sign-up fee.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 2-7 describes the steps for creating the discount sign-up fee.

Table 2-7 Defining the Product

Do This	Why
In the **Event Map** group, click **Add**. In the **Event** column, select **Product Purchase Fee Event**.	To create the sign-up fee by associating the rate details with the purchase event, as specified by the plan. When you select **Product Purchase Fee Event**, Pricing Center automatically selects **Occurrence** in the **Measured By** column. You cannot change this value. **Note:** The values in the **Rating Details** group are used only for time-dependent fees.
In the **Rate Plan Structure** column, select **Single Rate Plan**. Under **Single Rate Plan**, and click **Open Rate Plan**.	To set up the rates for this product.
In the **Plan Detail** tab, enter **purchase** in the **Name** field.	The rate plan name identifies the rate plan in the product and reminds you of its purpose. For system events, you must enter the exact name of the rate plan.
In the **Rate Structure** tree view, select **Tier 1**. In the **Tier Detail** tab, enter a name for the rate tier: **Discounted IP Sign-up Fee**.	The name identifies the rate tier in the product and reminds you that it is a discounted fee.
Under Effective Dates, select **Absolute**.	To make the rate available for a limited time, and to ensure that it expires on a certain date.
Under **Ends**, select **At**, then **On**, and set the end date to **03/01/2001**.	The discount is valid through the end of February, 2001. Valid date ranges do not include the end date; therefore, you enter the first date when this rate is no longer valid.
Under **Use Time/Day Restrictions**, select **No restrictions**.	To bypass setting valid days of the week.
In the **Rate Structure** tree view, select **Rate 1**. In the **Rate Data** tab, enter a rate name: **$7.50 before 3/1/01**. **Note:** Rate names must be unique. In the **/rate** storable class, the rate name is specified in the PIN_FLD_DESCR field.	To remind you of the purpose of this rate.
Select **Overrides credit limit**.	To continue to charge for Internet access when customers exceed the credit limit.
In the **Based on** list, select **Continuous**.	This option is the default when you use only one quantity discount bracket to rate an event.
In the **Rate Structure** tree view, select **No Minimum -No Maximum**.	This displays the **Balance Impacts** tab, where you define the cost of the rate.
Under **Valid Quantities**, select **None** for both the **Minimum** and **Maximum** fields.	The monthly fee is not based on a specific amount of the event.
Under **Balance Impacts**, click **Add**. **Tip:** Drag the right edge of any column heading to make the list wider.	To add a row to the impacts list and to specify the balance impacts **-** what you charge the customer for the service.
Click the **Resource ID** row and scroll to select **US Dollar [840]**.	To charge in US dollars.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 2-7 (Cont.) Defining the Product

Do This	Why
Click the **GLID** row and scroll to select **Purchase Fees [101]**.	To record the fees as purchase fees in your general ledger.
Select **S for Sponsorable**.	To enable the fee to be paid by another account. Select this option to enable one account to pay for another.
In the **Fixed Amount** field, type **7.5**.	One-time charges, such as the purchase fee in this product, are called *fixed* rates. You enter the dollar amount of the charge.
In the **Rate Structure** tree view, select **Purchase**.	You are finished specifying this rate tier.

Product 2a – Creating the Standard Sign-up Fee

You create the standard sign-up fee by using a second rate tier in the purchase rate plan. Creating the standard rate after creating the discounted rate gives the standard rate a lower priority, which ensures that the discount rate is used during the valid discount period.

Table 2-8 describes the steps for creating the standard sign-up fee.

Table 2-8 Creating the Standard Sign-up Fee

Do This	Why
In the **Rate Structure** tree view, select **Purchase** and then click **Add**.	To add a rate tier to the rate plan.
In the **Rate Structure** tree view, select **New Rate Tier**. In the **Tier Detail** tab, enter a name for the rate tier: Standard.	The name identifies the rate tier in the product and reminds you of the purpose of this rate.
Under **Tier Detail**, select **Absolute** or **Relative**, then **Starts Immediately** and **Never Ends**. Also, under **Use Time/Day Restrictions**, select No restrictions.	To make the rate available indefinitely, so it never expires in the BRM database.
In the **Rate Structure** tree view, select **New Rate**. In the **Rate Data** tab, enter a name for the rate: **$15 Sign-up Fee**. **Note:** Rate names must be unique. In the **/rate** storable class, the rate name is specified in the PIN_FLD_DESCR field.	To remind you of the purpose of this rate.
Select **Overrides credit limit**.	To continue to charge for the email account if customers exceed the credit limit.
In the **Based on** list, select **Continuous**.	This option is the default when you use only one quantity discount bracket to rate an event.
In the **Rate Structure** tree view, select **No Minimum - No Maximum**.	This displays the **Balance Impacts** tab, where you define the cost of the purchase rate.
Under **Valid Quantities**, select **None** for both the **Minimum** and **Maximum** fields.	The monthly fee is not based on a specific amount of the event.
Under **Balance Impacts**, click **Add**.	To add a row to the impacts list and to specify the balance impacts **-** what you charge the customer for the service.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 2-8 (Cont.) Creating the Standard Sign-up Fee

Do This	Why
Click the **Resource ID** box and scroll to select **US Dollar [840]**.	To charge in US dollars.
In the **GLID** list, select **Purchase Fees [101]**.	To record the charges as one-time purchase fees in your general ledger.
Click **S for Sponsorable**.	To enable the rate to be paid by another account. Select this option to enable one account to pay for another.
Click **D** for **Discountable**.	To enable the rate to be discounted when the service is purchased.
In the **Fixed Amount** field, type **15**.	One-time charges, such as the purchase fee in this product, are called “fixed" rates. You enter the dollar amount of the charge.
In **Units**, select **None [0]**.	This rate uses the **Occurrence** ratable usage metric (RUM), which is not measured in units.
Click **OK**.	This returns you to the Product Attributes dialog box. You are finished creating purchase fees. Now you must create the cycle fees.

Product 2a – Creating the Cycle Fees

You create the cycle fees by defining two rate tiers that map to the cycle forward monthly event. The first rate tier gives the customer the third month free, and the second rate tier charges the basic monthly fee of $19.95.

Product 2a – Creating the Free Month

You define the free month rate tier first to ensure that the free month rate is applied first instead of the standard monthly rate. You define the free month as a certain number of days from enrollment.

Table 2-9 describes the steps for creating the free month.

Table 2-9 Creating the Free Month

Do This	Why
In the **Event Map** group, click **Add**. In the **Event** column, select **Monthly Cycle Forward Event**.	To create the sign-up fee by associating the rate details with the purchase event, as specified by the plan.
In the **Measured By** column, select **Occurrence**. See ratable usage metric (RUM).	To specify that the fee is charged when the monthly cycle forward event occurs.
In the **Rate Plan Structure** column, select **Single Rate Plan**. Then, under **Single Rate Plan**, click **Open Rate Plan**.	To set up the rates for this product.
In the **Plan Detail** tab, enter **cycle_forward_monthly** in the **Name** field.	The rate plan name identifies the rate plan in the product and reminds you of its purpose.
In the **Rate Structure** tree view, select **Tier 1**. In the **Tier Detail** tab, enter a name for the rate tier: **Third Month Free**.	The name identifies the rate tier in the product and reminds you of when it is valid.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 2-9 (Cont.) Creating the Free Month

Do This	Why
Under **Effective Dates**, select **Relative**.	To make the rate available for a specific period relative to the product purchase date.
Under **Starts**, enter **55** and **Day**.	To specify that this rate starts 55 days after the product is purchased. After the customer pays for the first two months, BRM runs billing for the third month 59-62 days after registration. You want BRM to use the free month rate, so by specifying 55 days, you ensure that this rate is applied before BRM runs billing for the customer's third month.
Under **Ends**, enter **65** and **Day**.	To specify that this rate ends 65 days after the product is purchased. This is after BRM runs billing for the third month.
Under **Use Time/Day Restrictions** select **No restrictions**.	To bypass setting valid day ranges.
In the **Rate Structure tree view,** select **Rate 1. In the Rate Data tab,** enter a name for the rate: **No Monthly Fee**.	To identify the purpose of this rate tier.
Select **Overrides credit limit**.	To continue to bill for the email account if the customer exceeds the credit limit.
In the **Based on** list, select **Continuous**.	This option is the default when you use only one quantity discount bracket to rate an event.
In the **Rate Structure** tree view under the **No Monthly Fee** rate, select **No Minimum - No Maximum**.	This displays the **Balance Impacts** tab, where you define the cost of the cycle rate.
Under **Valid Quantities**, select **None** for both the **Minimum** and **Maximum** fields.	The monthly fee is not based on a specific amount of the event.
Click **Add**. **Tip:** Drag the right edge of any column heading to make the list wider.	To add a row to the impacts list and to specify the balance impacts **-** what you charge the customer for the service.
Click the **Resource ID** box and scroll to select **US Dollar [840]**.	To bill in US dollars.
In the **GLID** list, select **Monthly Fees [102]**.	To have the free month recorded as monthly fees in a general ledger accounting report.
For **Fixed** and **Scaled**, make sure the values are **0**.	You do not charge the customer for the third month.
In the **Rate Structure** tree view, click the **cycle_forward_monthly** rate plan.	You are finished creating this rate tier, and now must create the standard monthly fee.

Product 2a – Creating the Standard Monthly Fee

You define another rate tier to charge the $19.95 monthly fee. You create this rate tier second to ensure that the “free month" rate is applied first and instead of the standard monthly rate that one time.

Table 2-10 describes the steps for creating the standard monthly fee.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 2-10 Creating the Standard Monthly Fee

Do This	Why
In the **Rate Structure** tree view, select the **cycle_forward_monthly rate plan** and click **Add**.	To add a new rate tier to the existing cycle_forward_monthly rate plan.
In the **Tier Detail** tab, enter a name for the rate tier: **Monthly Access Fee**.	The name identifies the rate tier in the product and reminds you of when its valid.
Under **Tier Detail**, select **Absolute** or **Relative**, then **Starts Immediately** and **Never Ends**. Also, under **Use Time/Day Restrictions**, select **No restrictions**.	To make the rate always available and always charged when no other rates are valid.
In the **Rate Structure tree view,** select **New Rate. In the Rate Data tab,** enter a name for the rate: **$19.95 a Month**.	To identify the fee in this rate tier.
Select **Overrides credit limit**.	To continue to charge for the email account when the customer exceeds the credit limit.
In the **Based on** list, select **Continuous**.	This option is the default when you use only one quantity discount bracket to rate an event.
In the **Rate Structure tree view,** select **No Minimum - No Maximum.**	This displays the **Balance Impacts** tab, where you define the cost of the cycle rate.
Under **Valid Quantities**, select **None** for both the **Minimum** and **Maximum** fields.	The monthly fee is not based on a specific amount of the event.
Under **Balance Impacts**, click **Add**. **Tip:** Drag the right edge of any column heading to make the list wider.	To add a row to the impacts list and to specify the balance impacts **-** what you charge the customer for the service.
Click the **Resource ID** box and scroll to select **US Dollar [840]**.	To bill in US dollars.
In the **GLID** list, select **Monthly Fees [102]**.	To have the free month recorded as monthly fees in a general ledger accounting report.
In **Scaled Amount**, enter **19.95**.	Recurring fees, such as the monthly fee in this product, are called *scaled* rates. You enter the dollar amount of the fee.
Click **OK**.	You are finished creating this product.

Product 2b – Email Account, No Fees

The purpose of this product is to provide the customer with a single email account. Since the email account is included in the standard monthly fees for this plan, you do not charge the customer in this product.

Product 2b – Creating the Product

Table 2-11 describes the steps for creating the email account product.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 2-11 Creating the Product

Do This	Why
In Pricing Center, choose **Insert – Product** and select **Create a new product**.	To create a product.
Click **Next** and give the product a name: **Product 2b – Email Account, No Fees**. (You can also add a description.)	The product name identifies the product in Pricing Center. This name must be unique to all products in the same database brand.
When asked what type of product to create, select **Subscription** and in **Applies To**, choose **/service/email**.	This enables the product to contain recurring monthly rates and associates it with the email service.
Continue through the product creation wizard until you get to the **Valid Purchase Quantity** group.	Although you can change various attributes of the product while using the wizard, this sample product uses the default attributes up to this point.
Click **Maximum of** and enter **1**.	To specify that a customer can purchase no more than one email account.
Finish the product creation wizard, and click **Yes** at the end.	After you finish the wizard, you specify the rates for this product.

Product 2b – Creating the Free Rate

Even though the email account is free, you still must define an event rating map to track the free email service each month. You map the free email “fee" to the monthly cycle event.

Table 2-12 describes the steps for creating the free rate.

Table 2-12 Creating the Free Rate

Do This	Why
In the **Event Map** group, click **Add**. In the **Event Type** column, select **Monthly Cycle Forward Event**	To create the free monthly access fee by associating the rate details with the cycle event, as specified by the plan.
In the **Measured By column**, select the **Occurrence** ratable usage metric (RUM).	To specify that the rate is charged every month when the cycle forward event occurs. **Note:** The rating details are only used for time-dependent fees.
In the **Rate Plan Structure** column, select **Single Rate Plan.** Under **Single Rate Plan**, click **Open Rate Plan**.	To set up the rates for this product.
In the **Plan Detail** tab, enter **cycle_forward_monthly** in the **Name** field.	The rate plan name identifies the rate plan in the product and reminds you of its purpose. For system events, you must enter the predefined name of the rate plan.
In the **Rate Structure** tree view, select **Tier 1**. In the **Tier Detail** tab, enter a name for the rate tier: **Monthly Email Fee**.	The name identifies the rate tier in the product and reminds you when it is valid.
Under **Effective Dates**, select **Absolute**.	To set an indefinite period when the rate is available. **Note:** do not use **Always Applies**, because that sets a rate as a default rate, and charges the rate when no other rates apply. Since this rate is free, you do not want it to be the default rate.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 2-12 (Cont.) Creating the Free Rate

Do This	Why
Under **Use Time/Day Restrictions**, select **No restrictions**.	This rate is always valid; therefore, you do not need to specify which days it is valid.
In the **Rate Structure** tree view**,** select **Rate 1. In the Rate Data tab,** enter a name for the rate: **No Charges**. **Note:** Rate names must be unique.	To remind you of the purpose of this rate.
In the **Based on** list, select **Continuous**.	This option is the default when you use only one quantity discount bracket to rate an event.
In the **Rate Structure** tree view**,** select **No Minimum- No Maximum to display the Balance Impacts** tab.	This displays the **Balance Impacts** tab, where you define the cost of the rate.
Under **Valid Quantities**, select **None** for both the **Minimum** and **Maximum** fields.	The monthly fee is not based on a specific amount of the event.
Under **Balance Impacts**, click **Add**. **Tip:** Drag the right edge of any column heading to make the list wider.	To add a row to the impacts list and to specify the balance impacts **-** what you charge the customer for the service.
Click the **Resource ID** box and scroll to select **US Dollar [840]**.	To bill in US dollars.
In the **GLID** list, select **Monthly Fees [102]**.	To record the charges as monthly fees in your general ledger.
For **Fixed** and **Scaled**, make sure the values are **0**.	You do not charge the customer for the third month.
In **Units**, select **None [0]**.	This rate uses the **Occurrence** ratable usage metric (RUM), which is not measured in units.
Click **OK**.	You are finished creating the email product.

Deal 2a – Unlimited Internet Service

This deal contains the product for the sign-up fee and the Internet usage fees.

1. In Pricing Center, choose **Insert – Deal**.
2. Name the deal: **Deal 2a – Unlimited Internet Service**. (You can also add a description.)
3. In the **Applies to** list, select **/service/ip**.
4. In the **Products Associated with Deal** group, click **Add**.
5. In the **Product Name** list, select **Product 2a – Internet Access with Free Period**.
6. Click **OK** twice.

Deal 2b – Discounted Email Access

1. In Pricing Center, choose **Insert – Deal**.
2. Name the deal: **Deal 2b – Discounted Email Access**. (You can also add a description.)
3. In the **Applies to** list, select **service/email**.
4. In the **Products Associated with Deal** group, click **Add**.
5. In the **Product Name** list, select **Product 2b – Email Account, No Fees**.

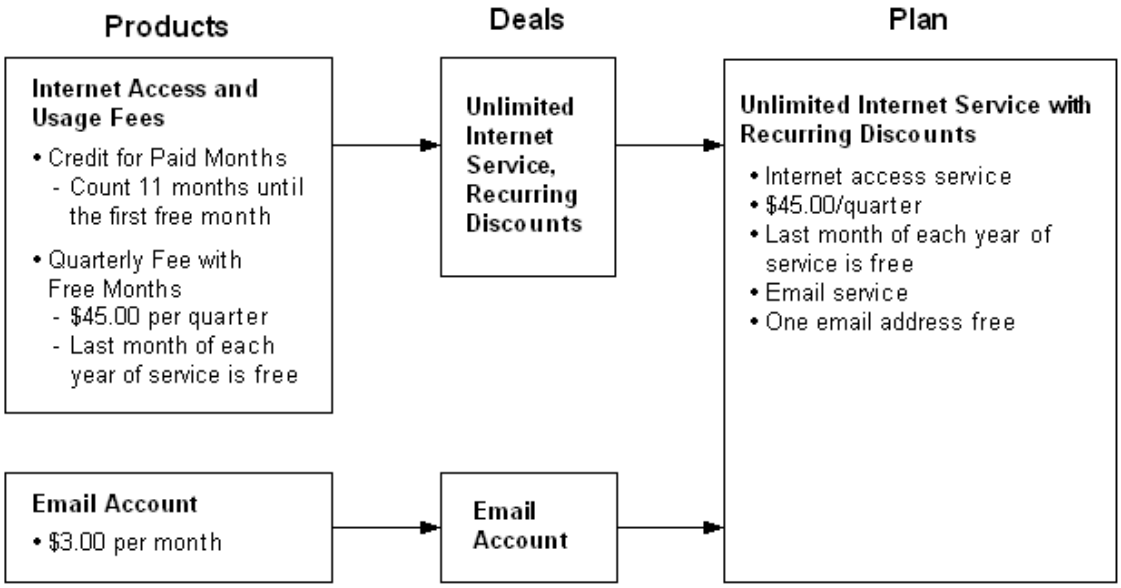
Copyright © 2025, 2026, Oracle and/or its affiliates.

6. Click **OK** twice.

Re-Creating Plan 3 – Unlimited Internet Service with Recurring Discounts

Figure 2-4 shows how to implement Internet service with recurring discounts.

Figure 2-4 Unlimited Internet Service with Recurring Discounts Plan



Putting together Plan 3

1. Create a custom BRM resource to use with this plan.

   BRM *resources* are units of exchange, such as dollars or hours, that are charged or credited when a customer uses your service. BRM comes with several common resources that you can use. For this plan, you must create one additional resource called “Paid Quarter". See "Creating a custom resource for Plan 3".

2. Create a product to charge for the Internet service based on which quarter of the year it is. See "Product 3a – Prepaid Internet Service with Free Period".

   > **Note**
   >
   > To offer the email account, you could create a second product specifically for Plan 3. However, Plan 2 has a product that already defines the email rate you need. You can use that product as a component of this plan, too.

3. Create a deal for Internet access. See "Deal 3 – Unlimited Internet Service, Recurring Discounts".

4. In Pricing Center, choose **Insert – Plan**.

5. Name the plan: **Plan 3 – Unlimited Web Access with Recurring Discounts**. (You can also add a description.)

6. In the **Services** section, click **Add**.

Copyright © 2025, 2026, Oracle and/or its affiliates.

7. For **Service Type**, select **/service/ip**.
8. For Deal, select **Deal 3a – Unlimited Internet Service, Recurring Discounts**.
9. Click **OK**.
10. In the **Services** section, click **Add**.
11. For **Service Type**, select **/service/email**.
12. For **Deal**, select **Deal 2b – Discounted Email Access**.

    As with products, you also can use a deal from another plan as a building block in this plan.
13. Click **OK**.
14. In the **Credit Limits** section, click **Add**.
15. In the **Resource ID** list, select **Paid Quarter**.
16. In the **Credit Limit** field, de-select **None**, then type **3**.
17. Click **OK**.
18. Click **OK** again to finish the plan.

Creating a custom resource for Plan 3

Table 2-13 describes the steps for creating a custom resource.

Table 2-13 Creating a Custom Resource

Do This	Why
From the start options, choose **Pricing Center**, which is under **Portal**, and then supply the login name and password to open the Pricing Center application.	You create resources with the Resource Editor. **Important:** Check with your BRM system administrator before adding resources to the BRM database.
Click **Resource Editor**.	You use the Editor to add new resources and modify existing resources.
Choose **Edit – Add New Item**.	To add the resource required by this plan.
In the **Resource ID** box, type **1000012**.	This number identifies the resource in BRM. When creating a resource, you can use any unique number between 1000001 and 4000000000. (Numbers below 1000001 are reserved for use by BRM.)
In the **Name** box, type **Paid Quarter**.	The name appears in the resource list in Pricing Center.
In the **Round To** box, select **0**.	BRM uses a whole number for the free month, so you do not need any rounding.
For **Status**, select **Active**.	BRM can only use active resources.
Click **OK**.	You are finished creating this resource.
Choose **File – Close** and **Commit New Changes**.	To add the new resource to the BRM database and close the Resource ID Editor.
Click **Yes** to confirm.	To confirm that you want to add the resource.

Product 3a – Prepaid Internet Service with Free Period

This product serves several purposes:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Charging the customer for 3 months of service at the beginning of each quarter.
- Counting the number of quarters the customer has been with the plan by tracking a “paid quarter" for each quarter. This is the resource you created for this plan.
- When the customer has remained with the plan for 3 quarters, this product credits the customer with $15 (the cost of one month of service) and resets the “paid quarter" resource to 0 to start the countdown for the next year.

Product 3a – Defining the Product

Table 2-14 describes the steps for defining the prepaid internet service with free period product.

Table 2-14 Defining Prepaid Internet Service with Free Period

Do This	Why
In Pricing Center, choose **Insert** – **Product** and select **Create a new product**.	To create a product.
Click **Next** and give the product a name: **Product 3a – Prepaid Internet Service with Free Period**. (You can also add a description.)	The product name identifies the product in Pricing Center. This name must be unique to all products in the same database brand.
Click **Next**. When asked what type of product to create, select **Subscription** and in **Applies To**, choose **/service/ip**.	This enables the product to contain usage rates and recurring monthly rates and associates it with the IP service.
Continue through the rest of the product creation wizard, and click **Yes** at the end.	Although you can change other attributes of the product while using the wizard, this sample product uses the remaining default attributes.

Product 3a – Creating the Quarterly Fee

This event rating map charges the customer for 3 months of service at the beginning of each quarter, and discounts the third quarterly fee by $15 to provide a free month of service.

It provides the discount by tracking one paid quarter for each quarter the customer has been with the plan. Then, when the customer reaches the third quarter, the rate is discounted $15 (the cost of one month of service) and the paid quarter balance is reset to 0 to start the countdown for the next year.

The rate uses event quantity ranges to determine the correct fee.

Table 2-15 describes the steps for determining the quarterly fee.

Table 2-15 Determining Fee

Do This	Why
In the **Event Map** group, click **Add**. In the **Event** column, select **Quarterly Cycle Forward Event**.	To specify billing at the beginning of each three-month cycle, as specified by the plan.
In the **Measured By column**, select **Occurrence**. See ratable usage metric (RUM).	To specify that the rate is charged every three months when the quarterly cycle forward event occurs. **Note**: The rating details are used only for time-dependent fees.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 2-15 (Cont.) Determining Fee

Do This	Why
In the **Rate Plan Structure** column, select **Single Rate Plan**. Under **Single Rate Plan**, click **Open Rate Plan**.	To set up the rates for this product.
In the **Plan Detail** tab, enter **cycle_forward_quarterly** in the Name field.	The rate plan name identifies the rate plan in the product and reminds you of its purpose. For system events, you must enter the predefined name of the rate plan.

Deal 3 – Unlimited Internet Service, Recurring Discounts

This deal bundles the product for the monthly counter, the product for the quarterly fees and free month, and the product for the email account.

1. In Pricing Center, choose **Insert – Deal**.
2. Name the deal: **Deal 3 – Unlimited Internet Service, Recurring Discounts**. (You can also add a description.)
3. In the **Applies to** list, select **/service/ip**.
4. In the **Products Associated with Deal** group, click **Add**.
5. In the **Product Name** list, select **Product 3a – Prepaid Internet Service with Free Period**.
6. Click **OK**.

Descriptions of the Advanced Sample Plans

This section describes each of the advanced sample plans including why certain pricing strategies were used to construct each plan. Each plan includes a description of what the plan does, key concepts illustrated by the plan, the customizations required to implement the plan, the structure of the plan, the target customer, and questions about the plan's structure and design.

ASP User Profile Plan

This plan (**ASPUserProfile.IPL**) provides an Enterprise Resource Planning (ERP) product with three categories of users: active, casual, and self-service. It charges $500 per month per active user, $150 per month per casual user, and $5 per month per self-service user. This plan charges a $22,000 minimum. After the per user charges reach $22,000, they begin to affect the customer's bill.

In addition, this plan provides volume discounts. If a customer is charged for more than $35,000 in a month, the charge for that month is discounted 10%. Similarly, if a customer is charged more than $40,000, the discount is 15%. If a customer is charged more than $50,000, the discount is 20%.

Concepts Illustrated

Real-time volume discounting and minimum charge.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Customization Required for the ASP User Profile Plan

The ASP User Profile plan includes the following custom resources:

- ERP Active (resource ID = 1000022)
- ERP Casual (resource ID = 1000023)
- ERP SelfService (resource ID = 1000024)
- Pre-discount Dollars (resource ID = 1000025)

You must add these custom resources to your BRM database if you intend to use the plan.

Adding Custom Resources

To add the custom resources:

1. Start the Configuration Center and connect to your BRM database.
2. Ensure that your database does not include custom resources with any of the resource IDs above: 1000022 through 10000025.

   If your database does include custom resources with those resource IDs, you must create the custom resources with different resource IDs and then change the plan to refer to those new resource IDs.
3. Add the resources.

Structure of the ASP User Profile Plan

The plan contains one product, ERP Product and six rate plans, add_active_user, add_casual_user, add_selfservice_user, del_active_user, del_casual_user, del_selfservice_user. In addition, this product contains one Fold rate plan selector that contains two rate plans, Fold Pre-discount dollars to US Dollars and Fold Royalty.

add_active_user rate plan

For each active user added, this rate plan increments the Pre-discount dollars resource by $500 and the ERP Active resource by 1.

add_casual_user rate plan

For each casual user added, this rate plan increments the Pre-discount dollars resource by $150 and the ERP Casual resource by 1.

add_selfservice_user rate plan

For each self-service user added, this rate plan increments the Pre-discount dollars resource by $5 and the ERP SelfService resource by 1

del_active_user rate plan

For each active user deleted, this rate plan decrements the Pre-discount dollars resource by $500 and the ERP active resource by 1.

del_casual_user rate plan

For each casual user deleted, this rate plan decrements the Pre-discount dollars resource by $150 and the ERP casual resource by 1.

Fold rate plan selector

Copyright © 2025, 2026, Oracle and/or its affiliates.

This rate plan selector contains two rate plans: **Fold Pre-Discount Dollars to US Dollars** and **Fold Royalty**.

The **Fold Pre-Discount Dollars to US Dollars** rate plan folds Pre-Discount Dollars into US Dollars resource balance based on the Pre-Discount Dollars resource balance. This rate plan contains five quantity discount brackets:

- The first quantity discount bracket increments the US Dollars resource by 22,000 if the Pre-Discount Dollars resource balance is under 22,000. This is, in effect, a minimum charge.
- The second quantity discount bracket applies if the balance of Pre-Discount Dollars is between 20,001 and 35,000. It multiplies the Pre-Discount Dollars resource balance by 1.00 and folds it into US Dollars.
- The third quantity discount bracket applies if the balance of Pre-Discount Dollars is between 35,001 and 40,000. It multiplies the Pre-Discount Dollar resource balance by 0.9 and folds it into the US Dollars resource balance. This is, in effect, a 10% discount for spending more than 35,000 Pre-Discount Dollars.
- The fourth quantity discount bracket applies if the balance of Pre-Discount Dollars is between 40,001 and 50,000. It multiplies the Pre-Discount Dollar resource balance by 0.85 and folds it into the US Dollars resource balance. This is, in effect, a 15% discount for spending more than 40,000 Pre-Discount Dollars.
- The fifth quantity discount bracket applies if the balance of Pre-Discount Dollars is greater than 50,000. It multiplies the Pre-Discount Dollar resource balance by 0.8 and folds it into the US Dollars resource balance. This is, in effect, a 20% discount for spending more than 50,000 Pre-Discount Dollars.

Questions About the Plan

Why is the Pre-Discount Dollars custom resource necessary?

The Pre-Discount Dollars custom resource is necessary to implement a minimum monthly charge and volume discounting. You track Pre-Discount Dollars instead and fold them into US Dollars at the end of the month to determine how the customer should be charged and which, if any, level of discounting should be given.

Audio/Video Plan

This plan (**AudioVideo.IPL**) gives 120 minutes of audio streaming and 60 minutes of video streaming per month for $14.95. The first month is free. Any audio minutes or video minutes remaining at the end of a month are lost.

After the first 120 minutes of audio streaming and the first 60 minutes of video streaming used in a month the plan charges for additional minutes based on time of day. From 8:30 a.m. to 6:00 p.m., audio is 25 cents a minute and video is 35 cents a minute. From 6:00 p.m. to 8:30 a.m., audio is 15 cents a minute and video is 25 cents a minute.

Concepts Illustrated

First month free, peak and off-peak hours, custom resources.

Customization Required for the Audio/Video Plan

The Audio/Video plan includes two custom resources:

- Minute Streamed Audio (Resource ID = 1000005)

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Minute Streamed Video (Resource ID = 1000006)

You must add these custom resources to your BRM database if you intend to use the plan.

Adding Custom Resources

To add the custom resources:

1. Start the Resource Editor and connect to your BRM database.
2. Ensure that your database does not already include custom resources with any of these Resource IDs: 1000005 or 10000006.

   If your database does include custom resources with those Resource IDs, you must create the custom resources with different Resource IDs and then change the plan to refer to those new Resource IDs.
3. Add the resources.

Structure of the Audio/Video Plan

The plan contains three products: AudioVideo Monthly, AudioVideo Audio Usage, and AudioVideo Video Usage.

AudioVideo Monthly Product

This product contains four rate plans. Two are single rate plans and two are combined in a rate plan selector.

The Monthly Cycle Forward Event rate plan contains an AudioVideo Monthly Rate Tier that charges $14.95 per month for the plan.

The Product Purchase Fee Event rate plan contains a Purchase Rate Tier that discounts the first month $14.95 when the plan is purchased. In effect, this makes the first month of service free.

The Cycle Fold Event rate plan selector combines two rate plans:

- The first rate plan, called Audio Fold, contains an Audio Fold Rate Tier that multiplies the balance of the Minute Streamed Audio resource by negative one (-1) and adds it to the resource balance. This resets the Minute Streamed Audio resource balance to zero.
- The second rate plan, called Video Fold, contains a Video Fold Rate Tier that multiplies the balance of the Minute Streamed Video resource by negative one (-1) and adds it to the resource balance. This resets the Minute Streamed Video resource balance to zero.

AudioVideo Audio Usage

This product contains one rate plan, IP Audio Event, with two rate tiers, Audio Basic Minutes and Audio Excess Minutes:

- The Audio Basic Minutes tier charges nothing for each minute of audio streamed up to 120 minutes per month.
- The Audio Excess Minutes tier charges a variable rate for each minute of audio streamed over 120 minutes per month. The rate structure includes two time of day periods: Peak and Off Peak. During Peak hours, 8:30 a.m. to 6:00 p.m., streamed audio is charged at 25 cents a minute. During Off Peak hours, 6:00 p.m. to 8:30 a.m., streamed audio is charged at 15 cents a minute.

Copyright © 2025, 2026, Oracle and/or its affiliates.

AudioVideo Video Usage

This product contains one rate plan, IP Video Event, with two rate tiers, Video Basic Minutes and Video Excess Minutes:

- The Video Basic Minutes tier charges nothing for each minute of video streamed up to 60 minutes per month.
- The Video Excess Minutes tier charges a variable rate for each minute of video streamed over 60 minutes per month. The rate structure includes two time-of-day periods: Peak and Off Peak. During Peak hours, 8:30 a.m. to 6:00 p.m., streamed video is charged at 35 cents a minute. During Off Peak hours, 6:00 p.m. to 8:30 a.m., streamed video is charged at 25 cents a minute.

Questions About the Plan

Why not just delay the initial monthly charge for one month instead of including a purchase product that refunds the first month's charge?

By including a purchase product that refunds the first month's charge, you can quickly change the plan to offer only a partial refund in the first month.

Customer Service Plan

This plan (**CustomerService.IPL**) is designed to let one company offer hosted customer service applications to a second company. The second company is then charged a fee for each subscriber to the service and is also charged for customer service minutes used by their subscribers.

The second company is charged a fee per subscriber and is also given a certain amount of customer service minutes per subscriber based on the total number of subscribers the second company has. The number of subscribers is checked when the Customer Service plan is first purchased and each time BRM billing is run - typically at the end of each month.

The charge per subscriber and the number of free customer service minutes given is determined as shown in Table 2-16.

Table 2-16 Subscribers and Charges

Number of Subscribers	Charge per Subscriber	Number of Free Customer Service Minutes per Subscriber
0 to 5000	$5	2
5000 to 10000	$4	3
10,000+	$3	4

Concepts Illustrated

Tiered pricing, products supporting business to business scenarios.

Customization Required for the Customer Service Plan

The Customer Service plan includes one custom service:

- **/service/customerservice**

Copyright © 2025, 2026, Oracle and/or its affiliates.

The Customer Service plan includes two different custom resources:

- Customer Service Minute (resource ID = 1000026)
- Subscription (resource ID = 1000502)

The Customer Service plan also includes two custom events:

- **/event/activity/add_subscription**
- **/event/activity/delete_subscription**

You must add these custom services, resources, and events to your BRM database if you intend to use the plan.

Adding Custom Services

You must use the Developer Workshop to add the **/service/customerservice** custom service to your BRM database.

Adding Custom Resources

To add the custom resources:

1. Start the Resource Editor and connect to your BRM database.
2. Ensure that your database does not include custom resources with any of the resource IDs above: 1000026 or 1000502.

   If your database does include custom resources with those resource IDs, you must create the custom resources with different resource IDs and then change the plan to refer to those new resource IDs.
3. Add the resources.

Adding Custom Events

This plan includes two custom events: **/event/activity/add_subscription** and **/event/activity/delete_subscription**.

You must use the Developer Workshop and edit **pin_event_map** to add these custom events to your BRM database.

Structure of the Customer Service Plan

The plan contains one product, Customer Service Product, with four rate plans, Add Subscription Event, Delete Subscription Event, Monthly Cycle Forward Event, Session Event, and one rate plan selector, Cycle Fold Event.

Add Subscription Event

This rate plan adds one to the Subscription resource when an **/event/activity/add_subscription** event occurs.

Delete Subscription Event

This rate plan deletes one from the Subscription resource when an **/event/activity/delete_resource** event occurs.

Monthly Cycle Forward Event

This rate plan has a customer service rate tier that both charges per subscription and allots free minutes based on the number of total subscriptions as shown in Table 2-17.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 2-17 Subscribers and Charges for Monthly Cycle Forward Event

Number of Subscribers	Charge per Subscriber	Number of Free Customer Service Minutes per Subscriber
0 to 5000	$5	2 (negative 2 added to the resource balance)
5000 to 10000	$4	3 (negative 3 added to the resource balance)
10,000+	$3	4 (negative 4 added to the resource balance)

Session Event

This rate plan has one rate tier, the Customer Service Usage Rate Tier, that adds one to the Customer Service Minute resource balance for each customer service minute used by the company's subscribers.

Cycle Fold Event

This rate plan selector has one rate plan, Fold Minutes. The rate plan folds the Customer Service Minute resource balance to US Dollars, $1 for each 1 minute used over the minutes given by the Cycle Forward Monthly Event rate plan. Then it folds the Customer Service Minute resource balance to zero by multiplying the balance by negative 1 and then adding it to the balance.

Note that the order in which the folds occur is very important. If the Customer Service Minute resource balance is folded first before the number of US Dollars is calculated, the company will never be charged.

Questions About the Plan

What happens if the number of subscribers during a month increases enough to qualify for a lower rate per subscriber and more customer service minutes per subscriber?

The rate tier is only checked when the plan is first purchased and each month when billing is run. For example, if a company starts a month with 4500 subscribers and then adds 700 subscribers during the month, they will still be charged per subscriber and given customer service minutes per subscriber based on having 4500 subscribers until the company runs billing again.

If all of the customer service minutes given to a company for the number of subscribers they have are not used by the end of the month, why does not the fold cause a credit to be issued to US Dollars?

The fold quantity bracket starts at “0" with no maximum so it is not activated if the customer service minute balance is negative. If you actually wanted a credit to be issued for customer service minutes not used, you could add another bracket to the fold that starts at no minimum and ends at “0".

IP Limited Access Plan

This plan (**IPLimitedAccess.IPL**) gives 100 hours of IP access for $19.95 per month. Each hour of IP access after a 100 hours is charged at $2.00 an hour. The first 12th month of IP service is free. Unused hours in each month are lost.

Copyright © 2025, 2026, Oracle and/or its affiliates.

For every IP access hour used, the customer gets 1 frequent flier credit at the end of the billing period. In addition, for every IP access hour in excess of 100 in a month, the customer is given 1 game credit which must be used by the end of that month.

The plan also includes unlimited email service for a one-time purchase fee of $5.00.

Concepts Illustrated

Free hours, free 12th month, hourly credits, folds.

Customization Required for the IP Limited Access Plan

The IP Limited Access plan includes several different custom resources:

- Bulk hours (Resource ID = 1000013)
- Paid month (Resource ID = 1000008)
- Frequent Flier Miles (Resource ID = 1000003)
- Free Game (Resource ID = 1000004)

You must add these custom resources to your BRM database if you intend to use the plan.

Adding Custom Resources

To add the custom resources:

1. Start the Resource Editor and connect to your BRM database.
2. Ensure that your database does not include custom resources with any of the Resource IDs above: 1000003, 1000004, 1000008, or 1000013.

   If your database does include custom resources with those Resource IDs, you must create the custom resources with different Resource IDs and then change the plan to refer to those new Resource IDs.
3. Add the resources.

Structure of the IP Limited Access Plan

The plan contains three products: IP Limited Access Email, IP Limited Access Monthly, and IP Limited Access Usage.

IP Limited Access Email Product

This is an item product consisting of a one-time charge of $5.00 for unlimited email service.

IP Limited Access Monthly Product

This is a subscription product containing two event types: Monthly Cycle Forward Event and Cycle Fold Event.

Monthly Cycle Forward Event

This includes a monthly rate tier with three quantity discount brackets.

- The first bracket charges $19.95 per month and increments the Paid Month resource by 1 each month.
- When Paid Months reaches 11, the second quantity discount bracket takes effect. The second bracket increments the Paid Month Resource by 1 but does not charge for that

Copyright © 2025, 2026, Oracle and/or its affiliates.

month. This bracket is valid only when Paid Month is equal to or greater than 11 and less than 12, so it gives the customer the 12th month of service free.

- The third quantity bracket takes effect after the 12th month of service. Since this bracket is valid when Paid Month is equal to or greater than 12, it remains in effect indefinitely after the 12th month of service. It affects resources the same way the first bracket does, thereby ensuring that only the first 12th month rather than every 12th month of service is free.

Cycle Fold Event

The Cycle Fold Event includes two fold rates which zero out the game credit balance at the end of the month and any hours of IP usage accrued in the month.

The folds zero out any remaining game credits and hours of IP service by multiplying the appropriate resource balance by -1 and adding it to the resource balance. For example, if a customer used 35 Bulk Hours of IP service and has 3 game credits left at the end of a month, the respective resource balances are 35 and 3. The fold rates multiply those remaining resource values by -1, resulting in -35 and -3 respectively, and adds those results to the resource balances to end with (35 + negative 35) or 0 Bulk hours and (3 + negative 3) or 0 Game Credits.

IP Limited Access Usage Product

This is a subscription product that includes one event type: Session Event.

Session Event includes two rate tiers.

- Base Rate Tier

  This rate tier is in effect until the Bulk Hours resource for a month reaches “100.00".

  In this tier, for each hour of service, the Bulk Hours resource is incremented by 1 and the Frequent Flier miles resource is incremented by 1. Thus, each hour decreases the number of base IP hours available and adds 1 frequent flier mile to the customer's account. The hours in the Base Rate Tier, the first 100 hours of IP usage in a month, have a balance impact of “0" because there is no usage fee for the first 100 hours in a month.

- Excess Rate Tier

  This rate tier takes effect after the first 100 hours of IP usage in a month.

  In this tier, for each hour of service, the customer is charged $2 and given 1 frequent flier mile and 1 game credit.

Questions About the Plan

How do I make every 12th month of base 100 hours of IP service free instead of just the first 12th month?

To make every 12th month free instead of just the first 12th month, change the quantity discount brackets within the Monthly Cycle Forward Event of the IP Limited Access Monthly product:

1. Leave the first quantity discount bracket as is.
2. Delete the third bracket.
3. Change the second quantity discount bracket as follows:
   a. Change the Maximum Valid Quantities to None.
   b. Change the Paid Month resource impact from 1.00 to (11.00).

Copyright © 2025, 2026, Oracle and/or its affiliates.

This zeroes out the Paid Month resource so the first bracket is used until the next 12th month of service is reached.

Since the number of Bulk Hours a customer uses is irrelevant as far as fees are concerned once they use over 100 in a month, why does the Excess Rate tier of IP Limited Access Usage continue to track the number of Bulk Hours used?

If Bulk Hours were not tracked by the Excess Rate tier, CSRs and customers would have no easy way of knowing how many excess hours the customer was charged for in a month. This way, the exact number of excess hours appears in Customer Center.

If Bulk Hours were not tracked, CSRs and customers must divide the excess charges by $2 to find the number of excess hours used in a month. But if excess hours were charged on a sliding scale by the plan, say $2 for the first 50 excess hours, $1.50 for the next 50 excess hours, and $.75 thereafter, the calculation would become more difficult and time consuming to perform. This way, the number of excess hours used is immediately available.

Online Articles Plan

This plan (**On-LineArticles.IPL**) charges $30.00 for quarterly access to an online articles service. The first five articles downloaded per month are free. The second five articles downloaded are charged at $2.00 an article. The next ten articles downloaded are charged at $1.50 an article. After that, all articles are charged at $1.00 an article.

In addition, if a customer downloads 30 or more articles in a month, the entire usage charge for that month is discounted 15%.

Concepts Illustrated

Quarterly cycle fee, variable usage fees based on monthly volume, discounting of all usage fees when a particular level of usage is reached.

Customization Required for the Online Articles Plan

The Online Articles plan includes two custom resources:

- Num Articles (resource ID = 1000015)
- Usage Charge (resource ID = 1000014)

You must add these custom resources to your BRM database if you intend to use the plan.

Adding Custom Resources

To add the custom resources:

1. Start the Resource Editor and connect to your BRM database.
2. Ensure that your database does not include custom resources with any of the resource IDs above: 1000014 or 10000015.

   If your database does include custom resources with those resource IDs, you must create the custom resources with different resource IDs and then change the plan to refer to those new resource IDs.
3. Add the resources.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Structure of the Online Articles Plan

The plan contains one product, On-Line Articles; two rate plans, Quarterly Cycle Forward Event, and Content Download Event; and one rate plan selector, Cycle Fold Event.

Quarterly Cycle Forward Event Rate Plan

This rate plan charges $30.00 per quarter. Note that the rate plan name is **cycle_fold_quarterly**.

Content Download Event Rate Plan

This rate plan includes four quantity discount brackets:

- The first quantity discount bracket increments the Num Articles resource by 1 for each article downloaded per month until 5 articles are downloaded.
- The second quantity discount bracket increments the Num Articles resource by 1 and increments the Usage Charge resource by 2.00 for each article downloaded per month until 10 articles are downloaded.
- The third quantity discount bracket increments the Num Articles resource by 1 and increments the Usage Charge by 1.50 for each article downloaded per month until 20 articles are downloaded.
- The fourth quantity discount bracket increments the Num Articles resource by 1 and increments the Usage Charge by 1.00 for each article downloaded over 20 per month with no maximum.

Cycle Fold Event Rate Plan Selector

This rate plan selector contains two rate plans:

- The first rate plan, Fold Usage Rate Plan, folds the Usage Charge into US Dollars based on the number of articles downloaded. If the Num Articles resource is less than 30, the Usage Charge resource balance is multiplied by 1.00 and folded directly into US Dollars. If the Num Articles resource is 30 or greater, the Usage Charge resource balance is multiplied by .85 and then folded into US Dollars. This, in effect, gives the user a 15% discount on all of their usage charges if they download 30 or more articles in a month.
- The second rate plan, Fold Articles Rate Plan, folds the Num Articles resource to zero by multiplying the balance by negative 1 and adding it to the balance.

Note that the order in which these rate plans are applied is very important. If the Fold Articles Rate Plan is applied first, customers will never get a 15% discount on their usage regardless of how many articles they download in a month.

You can control the order in which fold rate plans are applied by making sure the resource that you want to fold first either has a lower resource ID than the other resources you want to fold or is listed above the other resources you want to fold in the **pin_beid** file.

Questions About the Plan

Why is the Usage Charge custom resource necessary?

The Usage Charge custom resource is necessary because the 15% discount for 30 or more articles downloaded a month applies only to usage charges. Therefore you need a way of tracking usage fees separately from any other fees. If you simply charged US Dollars in the Content Download Event rate plan, you must either discount all customer fees, including any quarterly fees, or not discount customer fees at all.

Copyright © 2025, 2026, Oracle and/or its affiliates.

T1Access Plan

This plan (**T1Access.IPL**) includes a custom T1 access service. It provides the service for a monthly charge of $5,000 and an initial setup fee of $600. In addition, if the customer cancels the service before a year after the purchase date, they are charged a $10,000 cancellation fee.

Concepts Illustrated

Cancellation fee, custom service, relative date ranges for tiers

Customization Required for the T1 Access Plan

The T1 Access plan includes the custom service **service/T1.**

You must use the Developer Workshop to add this custom service to your BRM database.

Structure of the T1 Access Plan

The plan contains three products: T1 Purchase, T1 Monthly, and T1 Cancellation.

T1 Purchase Product

This product consists of a one-time purchase fee of $600 for starting the T1 service.

T1 Monthly Product

This product consists of a cycle forward event for $5,000 a month for the T1 service.

T1 Cancellation Product

This product consists of a one time cancellation fee of $5,000, triggered when the product is canceled. It is set up with a valid date range of 365 days. Thus, 365 days after the plan is purchased, the cancellation product and the associated cancellation fee no longer apply.

Questions About the Plan

Why is the cancellation fee a subscription instead of an item?

The cancellation event is generated only when the containing product is part of a subscription product. This is different from a purchase fee which is an item.

Where do you specify that the cancellation fee is valid for one year?

In the cancel rate tier, the rate tier duration is set as a Relative Date Range. The cancel rate is valid immediately and expires in 365 days.

This means that no matter when the T1 Access Deal is purchased, the cancellation fee will apply for 365 days after the purchase date.

Web Hosting Plan

This plan (**WebHosting.IPL**) charges customers for Web Hosting space. The customer is given 100 MB of Web Hosting space for $40 a month. If a customer is using more than 100 MB at the end of the month, they are sent a warning email. Each month after the first month that they exceed the limit, they are also charged 10 cents for every megabyte over 100 that have at the end of the month.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Concepts Illustrated

No usage charge up to a threshold, warning email, delayed charge for excess usage, bill excess over threshold.

Customization Required for the Web Hosting Plan

The Web Hosting plan includes the custom service **/service/diskspace/**, a custom event called Used More Disk Space, and two different custom resources:

- Warning (resource ID = 1000011)
- Peak MBs (resource ID = 1000503)

You must add this custom service, this custom event, and these custom resources to your BRM database for the plan to work.

Adding Custom Services

To add the **/service/diskspace/** custom service to your BRM database, use the Developer Workshop.

Adding Custom Events

To add the custom event Used More Disk Space, you must configure the DM, set the proper flag in Developer Workshop, map the event to the **/service/diskspace** service.

Note that the Used More Disk Space event must include custom code to set the Peak MBs resource correctly.

Adding Custom Resources

To add the custom resources:

1. Start the Resource Editor and connect to your BRM database.
2. Ensure that your database does not include custom resources with the resource IDs 1000011 or 1000503.

   If your database does include custom resources with those resource IDs, you must create the custom resources for the Web Hosting plan with different resource IDs and then change the plan to refer to those new resource IDs.
3. Add the resources.

Structure of the Web Hosting Plan

The plan contains one product, Web Host Product, with a rate plan, Monthly Cycle Forward Event, and a rate plan selector, Cycle Fold Event.

Monthly Cycle Forward Event Rate Plan

This rate plan consists of one tier, the Web Host Rate Tier, that charges $40.00 per month of Web hosting service.

Cycle Fold Event Rate Plan Selector

This rate plan selector includes two rate tiers, a Warning Tier and a Charge For Extra MB tier.

Copyright © 2025, 2026, Oracle and/or its affiliates.

The Warning Tier has two discount brackets. The first charges zero US Dollars if the Peak Megabytes resource balance is between 0 and 101 at the end of the month. The second increases the resource balance of the Warning resource by 1.00 if the Peak Megabytes resource balance is 101 or higher at the end of the month. This sends a warning message to the user that they have exceeded the 100 megabytes of storage offered by the plan.

The Charge for Extra MB Tier also has two discount brackets. The first charges zero US Dollars if the Peak Megabytes resource balance is between 0 and 101 at the end of the month.

The second discount bracket charges 10 cents for each megabyte used if the peak MBs used in a month is over 100. The discount bracket also credits the account $10, which is the cost of the first 100 megabytes at 10 cents a megabyte. This credit is issued so that you charge only fort he megabytes over 100.

The credit limit of the Warning Tier is set at 1.00 in the Web Host Plan. So the Charge for Extra MB Tier does not take effect until that credit limit is reached. This means the first month customers exceed the 100 MB threshold, they receive a warning email but are not charged for the excess usage.

Questions About the Plan

Why is there a quantity bracket tier in each of the Cycle Fold Event tiers that charges $0?

This is for readability. It is easier to see that there are no extra charges for the first 100 megabytes. You could remove the first tier and simply start at 101 megabytes of usage, but in an environment where price plans are often modified and used by multiple people, it is important to make the structure of the plan as clear as possible.

Why does the second discount bracket issue a credit of $10? Would it not be easier to code the Used More Disk Space custom event to track the number of megabytes over 100 used in a month?

This design enables you to quickly and easily change the number of megabytes allowed in a month. If you code the Used More Disk Space custom event to track the number of megabytes over 100 used in a month, you must update the code to change the number of megabytes allowed in a month rather than simply changing the pricing plan.

Copyright © 2025, 2026, Oracle and/or its affiliates.

3

Setting Up Real-Time and Pipeline Pricing and Rating

This chapter describes how to configure pricing data for the real-time rating and Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager.

Topics in this document:

- About Configuring Price List Data
- Setting Up Resources or Balance Elements
- Mapping Pipeline Manager Currencies and Real-Time Rating Currencies
- Setting Up Offer Profiles
- Setting Up Ratable Usage Metrics (RUMs)
- Mapping Event Types to RUMs
- About Mapping Services
- Mapping Service Codes and Service Classes
- Mapping Events and Services
- Mapping Usage Classes
- Mapping Usage Types
- Adding Customer Balance Impact Data to EDRs
- Modifying and Loading the EDR Container Description
- Dropping or Upgrading Incomplete Calls after Changing the EDR Container Description
- About Serviceless Accounts as Charge Sharing Owners
- Mapping Subscription Services in the Pipeline Manager Database
- Setting Up Batch Rating to Assign Items Based on Event Attributes

About Configuring Price List Data

Before creating your price list, you must do the following tasks to configure price list data:

- Create resources. See "Setting Up Resources or Balance Elements".
- Set up offer profiles. See "Setting Up Offer Profiles".
- Create ratable usage metrics (RUMs). See "Setting Up Ratable Usage Metrics (RUMs)".
- Map event types to RUMs. See "Mapping Event Types to RUMs".
- Specify which events you must rate. See "Mapping Events and Services".
- Map the external data to the internal usage class. See "Mapping Usage Classes".
- Define and map the usage types. See "Mapping Usage Types".

Copyright © 2025, 2026, Oracle and/or its affiliates.

Setting Up Resources or Balance Elements

Real-time rate plans and pipeline rate plans each use a different set of resources. See the following topics:

- Setting Up Resources for Real-Time Rating
- Setting Up Pipeline Manager Resources or Balance Elements

> **Note**
>
> You must use the same resource names for real-time and batch pipeline resources.

Setting Up Resources for Real-Time Rating

If you are using Pricing Center, you must use the Resource Editor in Pricing Center to create resources before you create a price list. You can create currency and noncurrency resources.

When you create resources, you define the following:

- The resource name, such as US Dollars.
- The resource ID, such as 840. Currency resource IDs are defined in an ISO standard.
- The rounding value. For example, to round US dollars to cents, specify a rounding value of two places after the decimal point.

> **Note**
>
> Resource Editor sets rounding only for A/R actions such as adjustments and refunds. To set rounding for rating, discounting, and taxation as well as A/R, configure rounding in the **pin_beid** file. See "Configuring Resource Rounding".

- How to round additional numbers beyond the rounding value:
  - **Up**: For example, 10.151 rounds to 10.16.
  - **Down**: For example, 10.159 rounds to 10.15.
  - **Nearest**: If the additional digit is 0-4, the last significant digit remains the same. If the additional digit is 5-9, the last significant digit is rounded up. For example, 10.144 rounds to 10.14 and 10.145 rounds to 10.15.
  - **Even**: If the additional digit is 0-4, the last significant digit remains the same. If the additional digit is 6-9, the last significant digit is rounded up. If the additional digit is 5, the last significant digit is rounded to the nearest even digit. For example, if rounding is set to two significant digits, 10.155 rounds to 10.16 and 10.165 rounds to 10.16.
- For currency resources, you also define the error tolerance and the abbreviations and symbols used for display, such as $.
- The order in which to consume resource sub-balances. For example, if a customer's balance contains free minutes with different validity periods, you specify which minutes to use first, according to the starting validity date or ending validity date.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Setting Up Pipeline Manager Resources or Balance Elements

If you are using Pricing Center or Oracle Communications Pricing Design Center (PDC), Pipeline Manager resources or balance element IDs must match other BRM resources or balance element IDs. For example, when defining pipeline charges, you use BRM resources or balance element IDs, such as US Dollars (840). Pipeline Manager resources or balance elements can be mapped to other BRM resources or balance elements.

Defining Currencies

You must define all valid currencies before setting up currency balance elements.

You define Pipeline Manager currencies in Pricing Center, Oracle Communications Pipeline Configuration Center (PCC), or PDC.

You must also configure the DAT_Currency module. This module converts currency symbols to numeric values and retrieves balance element rounding rules, using data from **/config/beid** objects in the BRM database.

Defining a Resource or Balance Element

Resources in Pricing Center or PCC and Balance Elements in PDC are arbitrary measurement values for which a separate charge can be calculated. You can define two types of balance elements: **Money** and **Other**. **Money** balance elements are based on a defined currency. **Other** balance elements are noncurrency balance elements, such as duration or loyalty points.

You define Pipeline Manager balance elements in Pricing Center, PCC, or PDC.

Defining Currency Exchange Rates

You must set up an exchange rate for each currency you support for rating. Exchange rates specify the rate for converting one currency into another.

Call details records (CDRs) can come from multiple networks in different countries, which use different currencies. When you use exchange rates, you can store up to three charge packets with different currencies in a single EDR. The currency types are:

- **Rating currency**: The currency defined in the charge and pricing.
- **Billing currency**: The currency associated with a specific customer, network operator, and so forth for billing and invoicing.
- **Home currency**: The Pipeline Manager system-wide currency.

You convert currencies from the rating currency to the billing currency, the home currency, or both.

You define exchange rates in Pricing Center or PDC. Each exchange rate includes the following data:

- The currency to convert from.
- The currency to convert to.
- The conversion rate.
- The date from which the conversion rate is valid.

To implement currency conversion, you configure the FCT_ExchangeRate module and the DAT_ExchangeRate module.

Copyright © 2025, 2026, Oracle and/or its affiliates.

When you configure FCT_ExchangeRate, you specify the exchange rate conversion mode by setting the **Mode** entry to Normal or Reverse. In Normal mode, the exchange rate is the multiplier to convert the From Currency to the To Currency. In Reverse mode, the exchange rate is the multiplier to convert the From Currency to the To Currency and the divisor to convert the To Currency back to the From Currency.

For example, suppose the exchange rate to convert from EUR to USD is defined as 2.0. One pipeline converts EUR to USD with **Mode** set to Normal and the another pipeline converts USD to EUR with Mode set to Reverse. In Normal mode, the exchange rate 2.0 is the multiplier to convert from EUR to USD (EUR * 2.0 = USD). In Reverse mode, the exchange rate 2.0 is the divisor to convert from USD to EUR (USD / 2.0 = EUR). If Reverse mode was not used, you would define another exchange rate 0.50 to convert from USD to EUR.

To update conversion rates, you use the DAT_ExchangeRate module **Reload** semaphore. The new exchange rates overwrite the old rates.

About Currency Exchange Validity Dates

You can use the FCT_ExchangeRate module to configure how to define the validity date for currency conversion for two cases.

- Use the **RatingDateBilling** registry entry to convert from the rating currency to the billing currency.
- Use the **RatingDateHome** registry entry to convert from the rating currency to the home currency.

You can specify a different value for the conversion to the billing currency and the home currency. For example, the validity date for converting the rating currency to the billing currency is usually the system time. The validity date for converting the rating date to the home currency is usually the CDR date.

For both registry entries, the values are:

- SYSTEM. The system time.
- FILE. The time that the file that includes the CDR was created (the file header time stamp).
- CDR. The time that the CDR was generated.
- NONE. (default).

> **Note**
>
> If you specify **RatingDateBilling** in addition to the **RatingDateHome** and **HomeCurrency** parameters, it converts the currency to the home currency only.

Mapping Pipeline Manager Currencies and Real-Time Rating Currencies

Currencies are stored as strings for Pipeline Manager and as numbers for real-time rating. You must map Pipeline Manager currency IDs to real-time rating currency names. To do so, you create a list of currencies in the FCT_BillingRecord module registry.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Setting Up Offer Profiles

If you are using Pricing Center, set up each offer profile in the following way:

> **Note**
>
> If you are using PDC, see the discussion about configuring policy specifications in *PDC Creating Product Offerings*.

1. Provide a unique name for the offer profile.

   > **Note**
   >
   > When setting up a product or a discount for an existing offer profile, use the offer profile name as the provision tag of the product or discount.
   >
   > Alternately, when you create an offer profile for an existing product or discount, use the provisioning tag of the product or discount as the name for the offer profile.

2. Configure a set of policy labels that define the currency or noncurrency resource and the quality of service in rate tiers.

   For example, you can define a policy label called *Fair Usage* for a noncurrency resource (such as *Megabytes Used*, whose resource Id is 100009). Set profile status levels based on predefined quality of service. The gradation may be as follows:

   - *Low QoS* for usage between 0 and 2.5 MBs
   - *Medium QoS* for usage between 2.5 and 5 MBs
   - *High QoS* for usage above 5 MBs

   Example 3-1 shows the contents of one sample policy label.

Example 3-1 Sample Policy Label

```
<policy_label>
   <label>Fair Usage</label>
   <resource_name>Megabytes Used</resource_name>
   <resource_id>100012</resource_id>
   <unit> 1 </unit>
   <tier>
       <tier_start_range>0</tier_start_range>
       <tier_end_range>2.5</tier_end_range>
       <status_label>High Qos</status_label>
       </tier>
    <tier>
       <tier_start_range>2.5</tier_start_range>
       <tier_end_range>5</tier_end_range>
       <status_label>Med Qos</status_label>
    </tier>
    <tier>
       <tier_start_range>5</tier_start_range>
       <status_label>Low Qos</status_label>
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
    </tier>
</policy_label>
```

Setting Up Ratable Usage Metrics (RUMs)

For background information about RUMs, see "About Ratable Usage Metrics".

Real-time rate plans and pipeline rate plans each use a different set of RUMs. See the following topics:

- About Setting Up RUMs for Real-Time Rating
- Setting Up Pipeline Ratable Usage Metric (RUM) Groups

About Setting Up RUMs for Real-Time Rating

If you are using Pricing Center, before you create your price list, you must define the RUMs available for rating. When you define RUMs, you define the following RUM attributes:

- The event type that can be rated by using the RUM (for example, Internet usage events). You can specify more than one RUM for an event type.
- The unit of measurement. For example, to rate duration, you might specify seconds or minutes. To measure bytes, you might specify megabytes or kilobytes.
- The RUM name (for example, “Duration," “Pages," or “Bytes"). The RUM name is displayed in Pricing Center.
- How to calculate the quantity. For example, to calculate the length of a session event, you subtract the time the event started from the time that the event ended:

  Event start: 7:00 p.m.

  Event end: 9:00 p.m.

  Event end - Event start = 2 hours

  This RUM is defined as follows:

  ```
  /event/session : Duration : PIN_FLD_END_T-PIN_FLD_START_T : second
  ```

  – The RUM name “Duration" appears in Pricing Center.
  – The duration is calculated by subtracting the start time from the end time.
  – The duration is calculated by using seconds.

To perform calculations, you can use simple arithmetic by using the following operators:

- +
- -
- *
- /
- (
-)

You can use three data types:

- PIN_FLDT_INT
- PIN_FLDT_DECIMAL

Copyright © 2025, 2026, Oracle and/or its affiliates.

- PIN_FLDT_TSTAMP

You can use fields in a substruct, but you cannot use arrays.

> **Note**
>
> The data used for the quantity calculation is stored in event object fields. For example, the event start and end times are stored in PIN_FLD_START_T and PIN_FLD_END_T. You must be familiar with events and event fields to specify quantity calculations.

To define the RUMs, you edit the **pin_rum** file and load it into the BRM database using the **load_pin_rum** utility. See "Creating Ratable Usage Metrics".

Creating Ratable Usage Metrics

To create RUMs, you edit the **pin_rum** file and then run the **load_pin_rum** utility to load the contents of the file into the **/config/rum** object in the BRM database.

> **Note**
>
> The **load_pin_rum** utility needs a configuration (**pin.conf**) file in the directory from which you run the utility.

To create ratable usage metrics:

1. Edit the **pin_rum** file in *BRM_home***/sys/data/pricing/example**. The **pin_rum** file includes examples and instructions.

   > **Note**
   >
   > - A RUM name can include a maximum of 255 characters.
   > - The **load_pin_rum** utility overwrites existing RUMs. If you are updating RUMs, you cannot load new RUMs only. You must load complete sets of RUMs each time you run the **load_pin_rum** utility.

2. Save and close the **pin_rum** file.
3. Use the following command to run the **load_pin_rum** utility:

   ```
   load_pin_rum pin_rum_file
   ```

   If you do not run the utility from the directory in which the file is located, you must include the complete path to the file. For example:

   ```
   load_pin_rum BRM_home/sys/data/pricing/example/pin_rum_file
   ```

   For more information, see "load_pin_rum".
4. Stop and restart the Connection Manager (CM). If necessary, stop and restart Pricing Center.

Copyright © 2025, 2026, Oracle and/or its affiliates.

To verify that the **pin_rum** file was loaded, you can display the **/config/rum** object by using Object Browser, or use the **robj** command with the **testnap** utility.

> **Note**
>
> Fold events cannot use custom RUMs; therefore, do not assign custom RUMs to fold events in any rate plans. Products configured with custom fold RUMs are rated incorrectly.

Setting Up Pipeline Ratable Usage Metric (RUM) Groups

RUMs are ways in which you can measure events, such as how long they last or how big they are. When you create RUMs, you give them names, such as Duration or Volume, and assign them a type, such as Duration, Event, or Normal.

Units of measurement (UoMs) are the names you give to units you will use to measure events, such as seconds or bytes.

A RUM group associates two or more RUMs together and associates a UoM with each RUM in the group.

When you rate events by using a RUM group, a separate charge packet is created for each RUM.

About Defining Units of Measurement (UoMs)

UoMs are the names you give to units you will use to measure events, such as seconds or bytes.

About Defining Ratable Usage Metrics (RUMs)

RUMs are types of event measurements that you define, such as duration or volume.

About Defining a RUM Group

A RUM group groups two or more RUMs together and associates a UoM with each RUM in the group.

Converting Units of Measurement

The UoM of an incoming EDR can be converted into a UoM needed for rating a particular service. For example, an EDR might include the usage amount in seconds, but the service is configured to charge by minutes. The FCT_UoM_Map module converts seconds to minutes, using rounding rules.

To configure UoM mapping:

1. Use Pricing Center or PCC to create a UoM map to convert UoMs. When you convert UoMs, you specify the following:
   - The RUM that uses the UoM.
   - The UoM to convert from.
   - The UoM to convert to.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- The conversion factor. For example, use a factor of 1024 to convert kilobytes into bytes.
- The rounding rule. The options are:
  - Nearest (N)
  - Always up (U)
  - Always down (D)

2. Configure the FCT_UoM_Map module.

Mapping Event Types to RUMs

If you are using Pricing Center for creating product offerings, you map event types to RUMs by using Pricing Center. However, you can edit the **pin_usage_map** file and load its content into the BRM database by running the **load_usage_map** utility. By doing so, you update the **/config/usage_map/system** object.

> **Note**
>
> - If you are using PDC, see the discussion about setting up the service-event map in *PDC Creating Product Offerings*.
> - The **load_usage_map** utility requires a configuration (**pin.conf**) file in the directory from which you run the utility.

To map event types to RUMs:

1. Edit the **pin_usage_map** file in *BRM_home***/sys/data/pricing/example**. The **pin_usage_map** file includes examples and instructions.

   > **Note**
   >
   > The **load_usage_map** utility overwrites existing usage maps. If you are updating a set of usage maps, you cannot load new usage maps only. You must load complete sets of usage maps each time you run the **load_usage_map** utility.

2. Save the **pin_usage_map** file.
3. Use the following command to run the **load_usage_map** utility:

   ```
   load_usage_map pin_usage_map_file
   ```

   If you are not in the same directory as the **pin_usage_map** file, include the complete path to the file. For example:

   ```
   load_usage_map BRM_home/sys/data/pricing/example/pin_usage_map_file
   ```

   For more information, see "load_usage_map".

4. Stop and restart the Connection Manager (CM). If necessary, stop and restart Pricing Center.

To verify that the data loaded correctly, display the **/config/usage_map/system** object by using Object Browser, or use the **robj** command with the **testnap** utility.

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Mapping Services

Incoming event data records (EDRs) from multiple switches often use different codes to represent the same service or supplementary service. To process EDRs, you must normalize that data by mapping external service codes to internal service codes, service classes, and usage classes. You also must map usage types so they can be added to EDRs.

- An internal *service code* represents a service such as voice or fax.
- An internal *service class* represents a variation on the service (for example, a GPRS quality of service (QoS) or a telephone service used only for emergency calls).
- An internal *usage class* represents a supplementary service (for example, call forwarding or a voice mail box). The data used for rating usage classes comes from the network.
- A *usage type* represents an attribute of a customer's account, such as an extended rating attributes (ERA), that is used in rating but which is not available from the network.

You define the internal codes and usage types in PCC. All of these are used to rate usage.

> **Note**
>
> If you are using PDC, see the discussion about setting up service and event definitions in *PDC Creating Product Offerings*.

To create internal service mappings, you use PCC to set up the data that specifies how to map the external data to the internal codes. For example, you can specify that if the external service code is **011**, the internal service code is **TEL**.

The mapping data you enter in PCC must match the syntax and format of the EDR input data.

About Creating Map Groups

You organize service codes, service classes, and usage classes in *map groups*. A map group specifies the set of services and usage scenarios to be used for rating by Pipeline Manager. You specify the name of the map group when setting up the pipeline modules.

You can create any number of map groups. When you have multiple map groups, you can use a different set of service mappings with different pipelines. For example:

- You rate services that use various EDR formats by setting up a pipeline for each format.
- If you rate roaming usage, you might set up three pipelines using these service mapping scenarios:
  - A retail rating pipeline that uses service mappings and usage scenarios relevant to your customer's accounts and services, such as friends-and-family calls, birthday calls, messaging services, and call forwarding.
  - Two pipelines (outcollect and incollect) for rating roaming usage that includes the services and usage scenarios described in your roaming agreements.

When you configure modules that use the map group, you enter the map group name in the registry. For example:

```
ServiceCodeMapping
{
  ModuleName = FCT_ServiceCodeMap
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
  Module
  {
    Active = True
    DataConnection = integrate.DataPool.Database
    MapGroup = serviceMapGroup
  }
}
```

> **Note**
>
> The map group name that you enter in the registry must exist in a map group configuration in the Pipeline Manager database.

Mapping Service Codes and Service Classes

To create service code and service class mappings, you use Pricing Center or PCC to specify which external data is used to determine the internal service codes and service classes. For example, you can specify that if the external service code is **011**, the internal service code is **TEL**.

The mappings are based on the following data:

- External service code
- Usage class
- Location area indicator
- VAS event charge offer code
- Quality of service requested
- Quality of service used
- Record type

The mappings are stored in the IFW_SERVICE_MAP table. The mapping is performed by the FCT_ServiceCodeMap module.

In addition to mapping external service codes to internal service codes, you also map internal Pipeline Manager service codes to service types defined in the BRM database. These mappings are stored in the IFW_SERVICE database table.

The configuration in Table 3-1 shows the mapping for three services:

Table 3-1 Service Mapping Configuration

Pipeline Manager Service Code	BRM Service Type
TEL	**/service/telco/gsm/telephony**
SMS	**/service/telco/gsm/sms**
GPRS	**/service/ip/gprs**

Multiple Pipeline Manager service codes can be mapped to a single BRM service type. For the best performance, use as few service mappings as possible. For example, map the TEL, SMS, DATA, and FAX Pipeline Manager service codes to a single **/service/telco/gsm** BRM service type.

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Setting Up Service Mapping

To set up service mapping, do the following:

1. Create internal service codes and service classes in Pricing Center or PCC.
2. Map the external data to the internal service codes and service classes in Pricing Center or PCC.
3. Configure the FCT_ServiceCodeMap module.

   This module maps call data to service codes by using the data you entered in Pricing Center or PCC.

Mapping Events and Services

When you create your product offerings, the usage events that you specify must be associated with services in the Pipeline Manager IFW_REF_MAP table. This table specifies the service type to event type mapping so that services are associated with usage events. In addition, the IFW_REF_MAP database table also specifies the prefix for the data that identifies an account (for example, MSISDN or IMSI). See "Configuring Account ID Prefixes".

You can map multiple Pipeline Manager service codes to a single BRM service type. For example, you could map the TEL, SMS, DATA, and FAX Pipeline Manager service codes to a single **/service/telco/gsm** BRM service type.

The data is stored in the Pipeline Manager database.

Mapping an Event Type to a BRM Service Type

You can map one event type to a single BRM service type (that is, one event type to a Pipeline Manager service code). For example, to rate GPRS and GSM usage events, you map the services as shown in Table 3-2:

Table 3-2 Mapping an Event Type to a Service Type

Service	Event
/service/telco/gsm/telephony	**/event/delayed/session/gprs**
/service/ip/gprs	**/event/delayed/session/telco/gsm**

You enter service-to-event mappings in Pricing Center or PCC.

Mapping Multiple Event Types to a BRM Service Type

To rate multiple events for a service, you map multiple event types to a single service type. For example, to rate GSM and GSMTEL usage events for GSM Telephony service, you map the **/event/delayed/session/telco/gsm** and **/event/delayed/session/telco/gsmtel event** types to the **/service/telco/gsm/telephony** BRM service type.

Rating multiple events for a service involves the following:

1. Enabling the BRM Database to Store Multiple Event Types per Service
2. Populating the Event Types in the EDR Container
3. Configuring Output Streams Based on Service Code and Event Types for iRules

Copyright © 2025, 2026, Oracle and/or its affiliates.

Enabling the BRM Database to Store Multiple Event Types per Service

To map multiple event types to a service type, you must update the one-to-one event type and service type mapping restriction in the BRM database. To do so, update the IFW_REF_MAP table.

By default, the IFW_REF_MAP table holds a PK_IFW_REM primary key constraint on the ID and the REF_OBJ columns. To update the one-to-one event type and service type mapping restriction in the BRM database, hold (or place) the PK_IFW_REM primary key constraint on the ID, REF_OBJ, and REF_PARAM columns.

Populating the Event Types in the EDR Container

When multiple event types are mapped to a service type, you must ensure that each event type is mapped to the appropriate service code.

You create an iScript or an iRule that will determine the correct event type for a service type in an EDR and populate the DETAIL.EVENT_TYPE field with the event type *before* passing the EDR through the following modules:

- FCT_Account module.
- FCT_AccountRouter module for the account router instance of the Pipeline Manager.

Configuring Output Streams Based on Service Code and Event Types for iRules

By default, the IRL_EventTypeSplitting iRule sends EDRs to separate output streams based on the service codes. When you configure multiple event types for a service type, you also map a service code to multiple event types. Therefore, you must update the IRL_EventTypeSplitting iRule and the data file to use the event types that are populated in the DETAIL.EVENT_TYPE field along with the service code to send EDRs to separate output streams.

To configure the output streams based on service codes and event types:

1. Update the IRL_EventTypeSplitting iRule to add a condition for checking the event types.

   The following IRL_EventTypeSplitting iRule sample shows the additional condition to check the event types:

   ```
   CONDITION:
   edrString(DETAIL.INTERN_SERVICE_CODE) =~ "${1}";
   edrString(DETAIL.EVENT_TYPE) =~ "${2}";
   ```

2. Edit the **IRL_EventTypeSplitting.data** file to include the event types. For example, the following entries map the GSM service code to the TELOutput and SMSOutput streams:

   ```
   GSM;/event/delayed/telco/gsm/telephony;TELOutput
   GSM;/event/delayed/telco/gsm/sms;SMSOutput
   ```

Example Procedure to Map Multiple Event Types to a Service Type

The following procedure is an example of how you map the **/event/delayed/session/telco/gsm/telephony** and **/event/delayed/session/telco/gsm/sms** event types to a single **/service/telco/gsm** service type.

1. Update the IFW_REF_MAP database table to hold (or place) the PK_IFW_REM primary key constraint on the ID, REF_OBJ, and REF_PARAM columns.

Copyright © 2025, 2026, Oracle and/or its affiliates.

2. This step is optional. In Pricing Center or PCC, map the **/service/telco/gsm** service type to two internal service codes, GSM1 and GSM2.
3. Map the **/event/delayed/session/telco/gsm/telephony** and the **/event/delayed/session/telco/gsm/sms** event types to the **/service/telco/gsm** service type.
4. Update **/config/event_map** to map the **/event/delayed/session/telco/gsm/telephony** and the **/event/delayed/session/telco/gsm/sms** event types to the **/service/telco/gsm** service type.
5. Create a charge offer with a **/service/telco/gsm** service type and configure the following two event types:
   - **/event/delayed/session/telco/gsm/telephony**
   - **/event/delayed/session/telco/gsm/sms**
6. Map the **/event/delayed/session/telco/gsm/telephony** and the **/event/delayed/session/telco/gsm/sms** event types to the required pipeline charge.
7. Create an iScript called **ISC_EventType.isc** to contain the logic that populates the DETAIL.EVENT_TYPE field with the appropriate event type.
8. Update the IRL_EventTypeSplitting iRule to map each combination of service code and event type to an output stream as follows:

```
GSM;/event/delayed/telco/gsm/telephony;TELOutput
GSM;/event/delayed/telco/gsm/sms;SMSOutput
GSM1;/event/delayed/telco/gsm/telephony;TEL1Output
GSM1;/event/delayed/telco/gsm/sms;SMS1Output
```

Mapping Usage Classes

To create usage class mappings, you use Pricing Center or PCC to set up the data that specifies how to map the external data to the internal usage class. For example, you can specify that if the external supplementary service code for call forwarding is **41**, the internal usage code is **CW**. In addition, you can use various EDR attributes to create variations on supplementary service for rating purposes. For example:

- **Service code mapping**. Use the usage class to create special virtual services and charge offers; for example, you can use the combination of telephony and call forwarding to create value-added services.
- **ServiceClass-Mapping**. Use the usage class to create special sub-services or quality of services (for example, service-level agreements).
- **UsageScenario-Mapping**. Use the usage class to map to impact categories.
- **RateAdjustment**. Use the usage class to provide discounts or adjustments in individual EDRs.

The mappings are based on the following data:

- External usage class
- Wholesale zone
- Tariff class and subclass
- Record type
- Connect type and Connect subtype, for example, anonymous, from another network, or direct
- Transit area code

Copyright © 2025, 2026, Oracle and/or its affiliates.

- APN address when you use GPRS or UMTS technology
- SS (supplementary service) packet

The mapping is performed by the FCT_UsageClassMap module. You can configure this module to use a specific map group.

About Setting Up Usage Class Mapping

To set up usage class mapping:

1. Create internal usage classes in Pricing Center or PCC.
2. Map the external data to the internal usage classes in Pricing Center or PCC.
3. Configure the FCT_UsageClassMap module.

   This module maps usage classes by using the data you entered in Pricing Center or PCC.

Mapping Usage Types

A usage type represents an attribute of a customer's account that is used in rating but which is not available from the network (for example, closed usergroup calls, friends-and-family calls, VIP calls, and birthday calls). A usage type is dynamically determined by comparison patterns using the A number and the B number.

You define usage types to represent the following customer information in an EDR:

- An ERA, such as an ERA for Friends and Family calls.

  The usage type used for rating ERAs is determined by the call origin and destination data in the EDR. For example, if the destination number in an EDR matches the number on the account's friends-and-family list, the corresponding friends-and-family usage type is added to the EDR and is used during rating.
- A group to which a customer belongs.

  This usage type is used by filter sets to apply system charge offers and system discounts to a select group of customers. For example, if the EDR's DETAIL.CUST_A.CUST_SEG_LIST field matches one or more values you define, the corresponding usage type is added to the EDR.

  > **Note**
  >
  > You must create your own application to add data to the CUST_SEG_LIST field.

The usage types that you define are available to all rating scenarios.

Usage types are added to EDRs by running the IRL_UsageType iRule. This iRule performs no rating, it just adds the usage type to the EDR DETAIL block. You specify the rules for mapping EDR data to usage types in the files used by the IRL_UsageType iRule.

About Setting Up Usage Type Mapping

To set up usage type mapping:

1. Configure the IRL_UsageType iRule files. See the following sections:
   - Configuring the IRL_UsageType iRule for ERAs

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Configuring the IRL_UsageType iRule for Filter Sets
- Configuring the IRL_UsageType iRule for Both ERAs and Filter Sets

2. Configure the FCT_IRule module. See "Configuring the FCT_IRule Module to Run the IRL_UsageType iRule".
3. Create usage types in Pricing Center or PCC.

Configuring the IRL_UsageType iRule for ERAs

The IRL_UsageType iRule uses the following files in the *Pipeline_home*/**iScriptLib** directory. The directory contains sample files.

- **ISC_UsageType.isc**. This file contains the iScript code that determines whether an EDR qualifies for a usage type. This script reads EDR fields and sets a series of variables. It then assigns a usage type to the EDR based on a mapping of variables to usage type.

  You can modify the **ISC_UsageType.isc** file to customize how to enter ERA data in Customer Center. For example, the following line uses a date format of *YYYY.MM.DD*:

  ```
  String strCallCombineYearMonth = "." + strCallMonth + "." + strCallDay;
  ```

  You can change this iScript to support a date format of *YYYY/MM/DD* by doing the following:

  ```
  String strCallCombineYearMonth = "/" + strCallMonth + "/" + strCallDay;
  ```

- **IRL_UsageType.irl**. This file contains the rules for mapping the variables to values. If the EDR contains an attribute that matches a variable, the value for that variable is set to **Yes**. Otherwise it is set to **No**.
- **IRL_UsageType.data**. This file contains the rules for assigning a usage type to the EDR. If the values of the variables match those specified in this file, the corresponding usage type is set in the EDR.

Configuration Example

> **Note**
>
> The sample configuration shows how to configure the iRule for ERAs only. It does not show how to configure the iRule for filter sets, which is done differently. For information, see "Configuring the IRL_UsageType iRule for Filter Sets" and "Configuring the IRL_UsageType iRule for Both ERAs and Filter Sets".

ISC_UsageType.isc

For ERAs, this file needs determination logic as shown in the following examples for the Friends and Family and Customer to Customer ERAs, respectively. This is in addition to the included **ISC_UsageType Function.isc** file, which is run when a condition is fulfilled.

```
isFriendsAndFamily = compareProductERAforProfileAttributes
(profileNameFriendsFamily,"NUMBER",
edrString(DETAIL.B_NUMBER), serviceType, EXACT_MATCH);

if  (edrNumDatablocks( DETAIL.CUST_A) > 0 and
edrNumDatablocks(DETAIL.CUST_B) > 0)
isCustomerToCustomer = "Y";
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

IRL_UsageType.irl

The following example shows the rules (conditions) for mapping variables to values for all ERAs. It also shows the result placeholder for the usage type. Each number corresponds to a position in the **IRL_UsageType.data** file.

```
CONDITION:
isRoaming =~ "${1}";
isInternational =~ "${2}";
isHomeRegion =~ "${3}";
isCustomerToCustomer =~ "${4}";
isSameClosedUserGroup =~ "${5}";
isSameCorporateAgreement =~ "${6}";
isSameCustomer =~ "${7}";
isSameSystemBrand =~ "${8}";
isSameSegment =~ "${9}";
isSameRateplan =~ "${10}";
isSameDiscountModel =~ "${11}";
isSameBillcycle =~ "${12}";
isBirthdayCall =~ "${13}";
isFriendsAndFamily =~ "${14}";
isHomeCell =~ "${15}";

RESULT:
edrString( DETAIL.USAGE_TYPE ) = "${16}";
```

IRL_UsageType.data

Each line in the **IRL_UsageType.data** file represents a different usage type mapping. Each position in a line corresponds to a variable (condition) in the **IRL_UsgeType.irl** file. The last position specifies the usage type code to apply if the mapping matches. This position corresponds to the result variable in the **IRL_UsageType.irl** file, which in the example above is “${16}";. Each position can have one of the following values:

- **N** = No
- **Y** = Yes
- **.** = any value

The following examples show the mapping and usage type for the Friends and Family and Customer to Customer ERAs, respectively.

```
N;N;.;.;.;.;.;.;.;.;.;.;Y;.;FNF

N;N;.;Y;.;.;.;.;.;.;.;.;.;.;.;CTC
```

Using the configuration examples above:

- If an EDR is not roaming or international (N in positions 1 and 2) and uses the Friends and Family ERA (Y in position 14), the usage type **FNF** is assigned.

  This mapping is associated with these conditions in the **IRL_UsageType.irl** file:

  ```
  isRoaming =~ "${1}";
  isInternational =~ "${2}";
  isFriendsAndFamily =~ "${14}";
  ```

- If an EDR is not roaming or international (N in positions 1 and 2) and uses the Customer to Customer ERA (Y in position 4), the usage type **CTC** is assigned.

  This mapping is associated with these conditions in the **IRL_UsageType.irl** file:

Copyright © 2025, 2026, Oracle and/or its affiliates.

```
isRoaming =~ "${1}";
isInternational =~ "${2}";
isCustomerToCustomer =~ "${4}";
```

Configuring the IRL_UsageType iRule for Filter Sets

The IRL_UsageType iRule uses the following files in the *Pipeline_home*/**iScriptLib** directory. There are samples of these files in that directory.

- **ISC_UsageType.isc**. This file contains the iScript code that determines whether an EDR qualifies for a usage type. This script reads EDR fields and sets a series of variables. It then assigns a usage type to the EDR based on a mapping of variables to usage type.
- **IRL_UsageType.irl**. This file contains a mapping to the contents of the **IRL_UsageType.data** file. For filter sets, this is a condition placeholder and a result placeholder.
- **IRL_UsageType.data**. This file contains one or more conditions and the usage type to be assigned. If the EDR contains attributes that match the conditions specified in this file, the corresponding usage type is set in the EDR.

Configuration example

> **Note**
>
> This sample configuration shows how to configure the iRule for filter sets only. It does not show how to configure the iRule for ERAs, which is done differently. For information, see "Configuring the IRL_UsageType iRule for ERAs" and "Configuring the IRL_UsageType iRule for Both ERAs and Filter Sets".

ISC_UsageType.isc

For filter sets, all you need is the included **ISC_UsageType Function.isc** file, which is run when a condition is fulfilled.

IRL_UsageType.irl

The following example shows a placeholder condition that indicates the position in the **IRL_UsageType.data** file that contains the determination logic for applying the usage type. This example also shows a result placeholder for the usage type.

> **Note**
>
> Do not include a string in the condition to represent the EDR field being checked. For filter sets, the fields are identified in the **IRL_UsageType.data** file.

```
CONDITION:
"${1}";

RESULT:
edrString( DETAIL.USAGE_TYPE ) = "${2}";
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

IRL_UsageType.data

Each line in the **IRL_UsageType.data** file represents a different usage type mapping. Each position in a line corresponds to the positions in the **IRL_UsageType.irl** file.

The following examples show rules for assigning a usage type to the EDR. For filter sets, the rule contains both the determination logic and the usage type to apply. You can use Boolean operators to define complex rules.

```
isSegmentContains("1234") and isSegmentContains("GOOD");A1

isSegmentContains("789") or isSegmentContains("OK");B2
```

Using the configuration examples above:

- If the CUST_SET_LIST field is set to **1234** *and* **GOOD**, the usage type **A1** is assigned.
- If the CUST_SET_LIST field is set to **789** *or* **OK**, the usage type **B2** is assigned.

Configuring the IRL_UsageType iRule for Both ERAs and Filter Sets

You can configure one set of IRL_UsageType iRule files for both ERA and filter set usage type mapping. For background information, see "Configuring the IRL_UsageType iRule for ERAs" and "Configuring the IRL_UsageType iRule for Filter Sets".

ISC_UsageType.isc

There is no special configuration in this file when you use it for both ERA and filter set usage type mapping. You need both the determination logic for ERAs and the included **ISC_UsageType Function.isc** file, which is run when a condition is fulfilled.

IRL_UsageType.irl

Combine both ERA variables and a placeholder condition for filter sets (${16}; in this example) in the **IRL_UsageType.irl** file. You need only one result entry.

```
CONDITION:
isRoaming =~ "${1}";
isInternational =~ "${2}";
isHomeRegion =~ "${3}";
isCustomerToCustomer =~ "${4}";
isSameClosedUserGroup =~ "${5}";
isSameCorporateAgreement =~ "${6}";
isSameCustomer =~ "${7}";
isSameSystemBrand =~ "${8}";
isSameSegment =~ "${9}";
isSameRateplan =~ "${10}";
isSameDiscountModel =~ "${11}";
isSameBillcycle =~ "${12}";
isBirthdayCall =~ "${13}";
isFriendsAndFamily =~ "${14}";
isHomeCell =~ "${15}";
${16};

RESULT:
edrString( DETAIL.USAGE_TYPE ) = "${17}";
```

IRL_UsageType.data

In the **IRL_UsageType.data** file, provide a separate line for each rule.

Copyright © 2025, 2026, Oracle and/or its affiliates.

The following is a line to assign a usage type for a filter set. The condition is in position 16 and the result is in position 17.

> **Note**
>
> Be sure to account for each condition in the **IRL_UsageType.irl** file.

```
.;.;.;.;.;.;.;.;.;.;.;.;.;.;isSegmentContains("1234") and isSegmentContains("GOOD");A1
```

The following is a line to assign a usage type for the Friends and Family ERA. The filter set position, 16, is set to any value (.).

```
N;N;.;.;.;.;.;.;.;.;.;.;Y;.;.;FNF
```

You can also combine ERA and filter set conditions in one rule. You can use this to apply an ERA usage type only to certain customers or to assign a filter set usage type to specific customers who also have an ERA:

```
N;N;.;.;.;.;.;.;.;.;.;.;Y;.;isSegmentContains("1234") and isSegmentContains("GOOD");C3
```

Configuring the FCT_IRule Module to Run the IRL_UsageType iRule

The IRL_UsageType iRule is run by the FCT_IRules module. You must run this instance of the FCT_IRules module after the FCT_Account module and before the FCT_USC_Map module.

Adding Customer Balance Impact Data to EDRs

> **Note**
>
> Before adding balance impacts to an EDR, you must configure internal service codes. See "About Mapping Services".

Before rating an EDR, you must add the customer's balance impact data to the EDR. This data includes the following:

- The POID of the service-level item that receives the balance impact.
- The start and end times of the accounting cycle that applies to the event. This data assures that the charges are applied to the correct bill item.

To find the balance impact data, you configure the following modules:

- Use the FCT_Account module to look up the balance impact data and add it to the EDR.
- Use the DAT_AccountBatch module to supply data to the FCT_Account module.

Configuring the DAT_AccountBatch Module

The DAT_AccountBatch module stores all customer data from the BRM database. The FCT_Account and FCT_AccountRouter modules use the data stored by the DAT_AccountBatch module.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> Because FCT_AccountRouter runs in a separate instance of Pipeline Manager, you configure separate instances of the DAT_AccountBatch module for the FCT_Account and FCT_AccountRouter modules.

The DAT_AccountBatch module stores the following types of data:

- *Dynamic data*, such as login names, charge offers, accounts, and services. This data is updated by the DAT_Listener module.
- *Maintenance data*, such as charge offers, packages, and bundles. Maintenance data is not updated constantly. Instead, it is updated only when all data is reloaded into the database.

To configure the DAT_AccountBatch module, see the following sections:

- Specifying Not to Load Closed Accounts
- Specifying Whether to Load All Account Data
- Specifying How Much Account Data to Retrieve on Startup
- Configuring Charge Offer Validity Checking
- Configuring Account Charge Offer Validity Checking for Backdated Events

Specifying Not to Load Closed Accounts

When you start Pipeline Manager, DAT_AccountBatch loads all accounts into memory. This can affect start-up performance when there are a great number of accounts. To improve Pipeline Manager start-up performance, you can reduce the number of accounts loaded into memory by specifying not to load closed accounts.

To not load closed accounts, you set the **ClosedAccountDelay** registry entry in the DAT_AccountBatch registry. Specify the number of days before the current date for which closed accounts are not loaded:

```
ClosedAccountDelay = number_of_days
```

For example, if the system time is 11:00 a.m. on June 25 and the number of days is 10, accounts that were closed before 11:00 a.m. on June 15 are not loaded into memory.

The **ClosedAccountDelay** registry entry applies only to closed accounts. Accounts that are inactive, and open accounts with inactive services, are still loaded into memory.

If delayed EDRs arrive in the pipeline for closed accounts that were not loaded into memory, those EDRs are rejected and not rated.

The **ClosedAccountDelay** registry entry uses only the system time, not the virtual time. If you set the virtual time and restart the pipeline, the virtual time is not considered when loading closed accounts.

For example, on June 20 you set the virtual time to June 5 and create an account. You then change the virtual time to June 9, close the account, and set **ClosedAccountDelay** to 10 days. When you restart the pipeline, the account is not loaded because it was closed more than 10 days before the system time (June 20), even though the current virtual time is June 9. EDRs that arrive for that account are rejected.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Specifying Whether to Load All Account Data

Use the DAT_AccountBatch module **InitialLoading** registry entry to specify whether the initial loading of service and account data is performed. If the data is not loaded, loading occurs while processing. Login objects are always loaded.

The default is **True**. Setting this entry to **False** enables the system to start faster, but processing might be slower.

Specifying How Much Account Data to Retrieve on Startup

Use the DAT_AccountBatch module **RowFetchSize** registry entry to specify the number of rows of data to retrieve from the BRM database. The default is 1000.

The value you enter is used only during the initialization of DAT_AccountBatch to improve performance. After the DAT_AccountBatch startup is completed, **RowFetchSize** automatically resets itself to a more appropriate value of 50 to conserve memory.

Configuring Charge Offer Validity Checking

You can use the DAT_Account module **UseProductCreatedTime** registry entry to configure how charge offer validity is checked:

- If you enable this entry, the charge offer created time is not considered when establishing charge offer validity. Only the start and end times are considered, so updates to charge offers are valid immediately.
- If you disable this entry, the module checks the created time and the start and end time of a charge offer when establishing the charge offer's validity. In some situations this procedure can lead to unexpected results. The created time (PIN_FLD_CREATED_T) is system-generated whenever the charge offer is modified. If the charge offer is modified after its start time (PIN_FLD_START_T), there can be a short period when the charge offer is invalid because its created time is later than the start time.

Configuring Account Charge Offer Validity Checking for Backdated Events

You can use the DAT_Account module **UseLatestProductAndDiscount** registry entry to configure validity checking for account charge offers and discount offers.

- If you enable this entry, when rating EDRs, DAT_AccountBatch retrieves account charge offer and discount offer information based only on their start and end dates.
- If you disable this entry, DAT_AccountBatch retrieves account charge offer and discount offer information based on their creation dates.

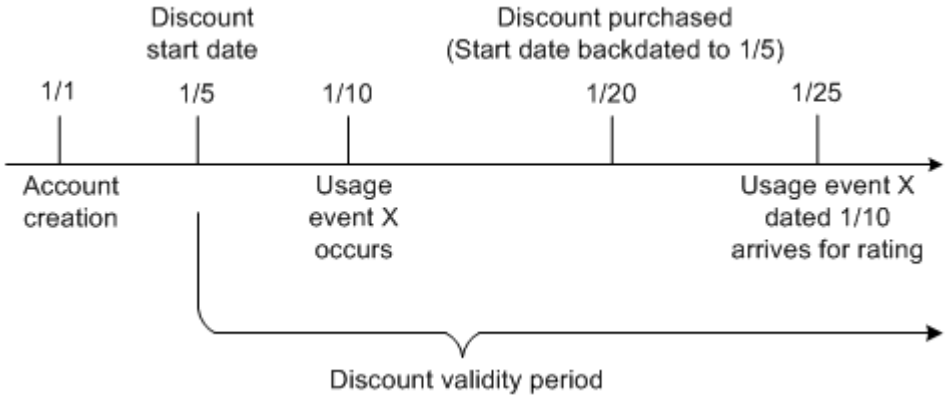
When charge offer and discount offer information is retrieved during EDR rating, pipeline modules validate the account charge offers and discount offers by using the account creation date. This date is the time that the charge offer and discount offer are purchased (the EFFECTIVE_T time in the AU_ACCOUNT_T audit table). If the charge offer's validity start date is earlier than the actual purchase date, which can occur when purchase events are delayed, usage events might be incorrectly rated.

For example, an account is created on January 1. On January 20, the customer purchases a discount offer with a validity period that starts on January 5. On January 25, a delayed event that is dated January 10 arrives for rating. The discount module does not apply the discount because it uses the purchase date of January 20 instead of the discount's start date of January

Copyright © 2025, 2026, Oracle and/or its affiliates.

5. Only events dated on or after January 20 are discounted. Figure 3-1 shows a timeline for this example.

Figure 3-1 Charge Offer Validity for Backdated Events Example



If you set the **UseLatestProductAndDiscount** entry in the DAT_AccountBatch registry, account charge offer and discount audit information is not retrieved. If you want delayed events to use the account charge offer and discount offer that was valid at the time the usage occurred, do not set this entry.

Getting Information about Loading Accounts

You can use DAT_AccountBatch semaphore commands to get data about the accounts loaded into the DAT_AccountBatch module.

- **PrintData** - Reports the account data for all accounts.

  The data is written to the specified file. If you do not specify a file, it is written to **stdout**.
- **PrintDataLogin** - Reports the account data for a single account identified by the BRM login ID (usually the phone number).

  The data is written to a file named *LoginID*.**lst**.
- **PrintDataSamples** - Reports the account data for a specified number of accounts, chosen randomly.

  The data is written to a file named *Value*.**lst**. For example, if you enter **15**, the file is named **15.lst**.

  If no value is provided, the module prints data for 10 accounts to **stdout**.

Configuring the FCT_Account Module

To find the balance impact data:

1. The FCT_Account module looks for the following data in the EDR:
   - The internal service code that indicates which data can be used to identify the account that generated the EDR. For example, if the internal service code is a telephony service, the identifying data is the A number. A different service might use the IMSI as the identifier.

     You identify which data to use by using Pricing Center or PCC. See "Specifying Which Data Is Used for Identifying Accounts". Depending on the service, the module

Copyright © 2025, 2026, Oracle and/or its affiliates.

searches for either the login from a service object or an alias from a service object. An alias is typically a phone number or IMSI.

In rare cases, one customer's MSISDN can be the same as another customer's IMSI. You can configure account ID prefixes to use for handling duplicate telephone numbers. See "Configuring Account ID Prefixes".

- The timestamp for the EDR. The timestamp is important because telephone numbers can be used by different accounts at different times.

2. The FCT_Account module uses the DAT_AccountBatch module to look up the account.

> **Note**
>
> - If the A customer is not found, the EDR is rejected. If the B customer is missing, no error is generated.
> - Because phone numbers can be recycled, the search is made on data from BRM audit objects.
> - Accounts are loaded based on the service that is being rated by the pipeline. If an account does not own the service that the pipeline rates, it is not loaded.

3. The DAT_AccountBatch module returns the balance impact data.
4. The FCT_Account module inserts the customer data into the EDR.

Specifying Which Data Is Used for Identifying Accounts

Different services require different ways to look up accounts. For example, to look up a telephone account, you can use the A number to find the originating account and the B number to find the account that was called.

To specify which data to use for looking up an account, you use Pricing Center or PCC to map the service to the type of data. The data is stored in the IFW_ALIAS_MAP table.

You can set up multiple mappings to support different services (for example, specify different mappings for each pipeline).

> **Note**
>
> This data is used by both the FCT_Account module and the FCT_AccountRouter module.

Use the following data to specify how to identify accounts:

- The EDR container description that includes the field to use for identifying the account.
- The account reference. You must use one of the following:
  - Use **Account_CustA** for the A number.
  - Use **Account_CustB** for the B number.
- The internal service code (for example, **TEL** or **DATA**).
- The type (for example **Internal** or **Plugin**).

Copyright © 2025, 2026, Oracle and/or its affiliates.

Configuring Account ID Prefixes

In some cases, the data used for identifying an account is not unique; for example, the MSISDN of one customer might have the same value as the IMSI of another customer. To avoid any conflicts, you can specify a prefix for the identifying data. This prefix data is stored in the IFW_REF_MAP database table.

You can specify a prefix based on the service type and event type, as shown in Table 3-3. In this case, the number is prefixed with either **msisdn** or **imsi**.

Table 3-3 Prefix Configurations

Service	Event	Prefix	Example of Result
/service/telco/gsm/telephony	/event/delayed/session/gprs	msisdn	msisdn00491721234567
/service/ip/gprs	/event/delayed/session/telco/gsm	imsi	imsi00491721234567

To set up prefixes:

1. Customize the PCM_OP_NUM_POL_DEVICE_ASSOCIATE and PCM_OP_SIM_POL_DEVICE_ASSOCIATE policy opcodes to add a prefix to the MSISDN or IMSI.
2. In the Pipeline Manager IFW_REF_MAP object, use the IFW_REF_MAP.REF_COL attribute to specify the prefix, for example, **msisdn**.

Specifying Which Noncurrency Sub-Balances to Load on Startup

The DAT_BalanceBatch module uses the noncurrency balance element validity to select the noncurrency sub-balances to load from the BRM database into pipeline memory. For example, if the Free Minutes balance element validity is 30 days, at Pipeline Manager startup, DAT_BalanceBatch selects all Free Minutes sub-balances that were valid for the last 30 days.

You configure the noncurrency balance element validity in the balance element configuration file (**pin_beid**). If the noncurrency balance element validity is not configured, DAT_BalanceBatch selects the sub-balances that were valid for 366 days by default

> **Note**
>
> When you set the noncurrency balance element validity, the sub-balances that do not fall within the validity range are not loaded into pipeline memory and therefore are not available for rating or rerating of call details records (CDRs). To make available the sub-balances that are required for rating and rerating, set the balance element validity to an appropriate setting. For instance, if the CDRs are older than 366 days, you should set the validity greater than 366 days to ensure that all the sub-balances that are required are loaded and available for rating or rerating the CDRs. See "Configuring the Noncurrency Balance Element Validity".

If your business requires, you can configure DAT_BalanceBatch to load all the noncurrency sub-balances available in the BRM database. To load all the noncurrency sub-balances, use the **SelectiveSubBalLoad** entry in DAT_BalanceBatch registry. When this entry is set to

Copyright © 2025, 2026, Oracle and/or its affiliates.

False, DAT_BalanceBatch loads all noncurrency sub-balances, including the sub-balances that are expired, into pipeline memory.

> **Note**
>
> When the BRM database contains a large number of noncurrency sub-balances, loading all sub-balances leads to increased Pipeline Manager startup times.

Configuring the Noncurrency Balance Element Validity

To configure the validity of a noncurrency balance element:

1. Edit the **pin_beid** file in *BRM_home*/**sys/data/pricing/example**.

   Follow the instructions in the **pin_beid** file.

   > **Note**
   >
   > If you are using an existing **pin_beid** file, you must manually add the **Validity_in_days** field to the syntax and set the balance element validity for the noncurrency balance element. In this example, the balance element validity for balance element **1000010** is set to **30** days.
   >
   > ```
   > #beid r_mode round tol_min tol_max tol_pct beid_str_code symbol event stage
   > con_rule_id apply_mode Name Validity_in_days
   > #
   > 1000010 0 1 0.000000 0.000000 0.000000 MIN Min * 0 0 1 Free Domestic Minutes
   > 30
   > ```

   > **Note**
   >
   > The **load_pin_beid** utility overwrites the existing balance elements. If you are updating balance elements, you cannot load new balance elements only. You must load complete sets of balance elements each time you run the **load_pin_beid** utility.

2. Save and close the **pin_beid** file.
3. Run the following command:

   ```
   load_pin_beid pin_beid
   ```

   If you do not run the utility from the directory in which the **pin_beid** file is located, you must include the complete path to the file. For example:

   ```
   load_pin_beid BRM_home/sys/data/pricing/example/pin_beid
   ```


Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Tip**
>
> If you copy the **pin_beid** file to the directory from which you run the **load_pin_beid** utility, you do not have to specify the path or file name. The file must be named **pin_beid**.

4. Stop and restart the Connection Manager (CM).

   To verify that the noncurrency balance elements were loaded, you can display the **/config/beid** object by using Object Browser, or use the **robj** command with the **testnap** utility.

5. Start Pipeline Manager.

   At Pipeline Manager startup, DAT_BalanceBatch uses the noncurrency balance element validity in the **/config/beid** object to select the noncurrency sub-balances to load into pipeline memory.

Modifying and Loading the EDR Container Description

By default, BRM supports the following EDR container description:

- **containerDesc.dsc:** The *rating EDR container description*. Defines the format into which CDRs are converted so that they can be processed for rating by a pipeline input module.

The EDR container description is in the *Pipeline_home***/formatDesc/Portal** directory.

If the CDR requests you process include information that has no corresponding fields in the appropriate default EDR container description, you must modify the description to accommodate the custom data. See "Modifying EDR Container Descriptions".

After editing the rating EDR container description, you must load it into the appropriate Pipeline Manager database. See "Loading EDR Container Descriptions".

Modifying EDR Container Descriptions

If the CDR requests you process include information that has no corresponding fields in the default EDR container description, you must modify the description to accommodate the custom data.

The rating EDR container description (**containerDesc.dsc**) text file is located in *Pipeline_home***/formatDesc/Portal**.

EDR Container Description Format

An EDR container description has the following format:

```
ComplexDataType
{
  String             StringVar;  //description
  Integer            IntegerVar;
  Date               DateVar;
  Decimal            DecimalVar;
  ComplexDataType_2  complexVar;
}
ComplexDataType_2
{
  String             var1;
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
  Decimal           var2;
}
```

The following example shows how to format an EDR container description file:

```
// Comments
DETAIL // comment
{
String A_Number; // Comment will be filled into the ifw_edrc_field.description
Date STARTDATE;
Decimal duration;
CUSTOMER CUST_A; // New complex type must be defined later on
CUSTOMER CUST_B;
}
HEADER
{
// comment
String name;
Integer count;
}
CUSTOMER
{
String account;
String account_no;
Purchased_Product PRODUCT;
}
Purchased_Product
{
String name;
Integer type;
}
```

Note the following syntax and grammar rules:

- The file is case-sensitive.
- Real data types are the following:
  - String
  - Integer
  - Date
  - Decimal
  - Block (for example, DETAIL.CUST_A)
- The parsing is done in one step. Therefore, embedded complex data types must be defined later in the file.
- The name of each complex data type must be written in one line.
- The complex data type area must begin with a left brace ({) and end with a right brace (}). Each brace must be in its own line.
- Each data type variable pair must be written in one line. The line must end with a semicolon (;).
- Comments must begin with double slashes (//) and are valid until the end of the line.
- Comments following a real data type line are copied to the IFW_EDRC_FIELD.DESCRIPTION field.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Loading EDR Container Descriptions

After editing the rating EDR container description, you must load it into the Pipeline Manager database.

You use the **containerDescLoader.pl** utility to load an EDR container description into a database.

EDR container descriptions are stored in the following database tables:

- IFW_EDRC_DESC
- IFW_EDRC_FIELD

Note the following restrictions:

- The Container Description Loader utility can process only one EDR container description file at a time.
- You cannot update tables that already include data. You can initialize empty tables or delete the data from tables before adding new data. You cannot, however, delete existing data if there are references to it from elsewhere in the database.

Creating the EDR Load Utility Command File

The Container Description Loader utility uses a command file to load data into a database.

The syntax of the command file is as follows:

```
#comments
ECHO print this on default output
TYPE=Database type
SOURCE=Database source
USER=Database user
PASS=Database user password
INIT=TRUE | FALSE
EDRC_DESC=Pipeline name as specified in the registry
EDRC_DESC_FILE=Container description file
EDRC_DESCRIPTION=Description of pipeline name
RUN=Execute this command
```

This is a sample command file with only the required entries:

```
DB=PR01|zaphod|zaphod
EDRC_DESC_FILE= /FMD/Portal/containerDesc.dsc
EDRC_DESC='MY_IFW'
EDRC_DESCRIPTION='MY_IFW container description'
```

This sample includes optional entries:

```
##
## This is a sample command file for containerDescLoader.pl
##
##
## DB (database connect)
## database|user|password
##
DB=PR01|zaphod|zaphod
#
# initialization (deleting tables)
#
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
ECHO=###################################
ECHO=# deleting of ifw_edrc_desc and ifw_edrc_field
ECHO=###################################
#INIT=FALSE
INIT=TRUE
# container description file
EDRC_DESC_FILE= /FMD/Portal/containerDesc.dsc
# pipeline name as specified in the registry
# filled into IFW_EDRC_DESC.EDRC_DESC and IFW_EDRC_FIELD.EDRC_DESC
EDRC_DESC='MY_IFW'
# additional description
# filled into IFW_EDRC_DESC.NAME
EDRC_DESCRIPTION='MY_IFW container description'
# optional command processing
#RUN=prog
```

Before Running the EDR Load Utility

Before running the Container Description Loader utility, do the following:

- Ensure that the following components are installed:
  - Perl 5.004 or higher
  - CPAN generic database interfaces
  - **oraperl** Perl module
- If necessary, create the following database tables:
  - IFW_EDRC_DESC
  - IFW_EDRC_FIELD
- If necessary, create the following sequence:
  - IFW_SEQ_FIELD_ID
- Use the **IFW_PERL_LIB** variable to find the required Perl modules.

Starting the EDR Load Utility

To start the Container Description Loader utility:

1. Go to the **TLS_ContainerDescLoader** directory.
2. Enter the following command:

   containerDescLoader.pl [**-c** *command_file*].

Dropping or Upgrading Incomplete Calls after Changing the EDR Container Description

The FCT_CallAssembling module stores any incomplete call records while a pipeline is running. When you stop the pipeline and change its container description, these incomplete call records are then automatically in the wrong format for the pipeline to process. This section describes your two choices for processing these call records:

- Discard them. This is the fastest way of dealing with the problem, but you lose the revenue they represent. You can however, use these EDRs for auditing purposes. For the procedure, see "Discarding Incomplete Calls after Changing the EDR Container Description".

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Upgrade them to the new container description. You use the tools provided to create a data upgrade pipeline that reformats the EDRs into the new container description format. For the procedure, see "Upgrading Incomplete Calls to the New Container Description".

These sections apply to all pipelines using FCT_CallAssembling to assemble CDRs into EDRs for the pipeline to process.

Discarding Incomplete Calls after Changing the EDR Container Description

This section explains the steps necessary to delete your partially-assembled call records during the process of changing the EDR container description.

You use the **UpgradeFlushLimit** semaphore registry entry to FCT_CallAssembling to flush and discard any incomplete calls stored in a pipeline during an EDR container description change.

You then use the **EmitPartialEDROnUpgrade** startup registry entry to specify what happens to the EDRs flushed by the **UpgradeFlushLimit** semaphore:

- If the **EmitPartialEDROnUpgrade** registry entry is **False**, the EDRs are silently dropped and are never processed by the pipeline.
- If the **EmitPartialEDROnUpgrade** registry entry is **True**, partial EDRs are processed by the pipeline but are not rated. You can use these records for revenue assurance purposes, but they should be discarded to avoid recycling.

EDR Data Available for Auditing

By default, all flushed EDRs will contain the following data fields:

- CHAIN_REFERENCE
- LONG_DURATION_INDICATOR (set to **P**)
- NUMBER_OF_CDRS
- CHARGING_START_TIMESTAMP
- CHARGING_END_TIMESTAMP
- DURATION

The following additional fields will also be included if the FCT_CallAssembling module has **AssembleVolume** set to **True**:

- VOLUME_SENT
- VOLUME_RECEIVED
- NUMBER_OF_UNITS
- RETAIL_CHARGED_AMOUNT_VALUE
- WHOLESALE_CHARGED_AMOUNT_VALUE

All EDR fields listed in the **AssembledEDR** registry entry are included if the **L** segment has been received.

Upgrading Incomplete Calls to the New Container Description

This section explains the steps necessary to update your partially-assembled call records to a new EDR container description. These tasks are necessary to correctly process any calls that

Copyright © 2025, 2026, Oracle and/or its affiliates.

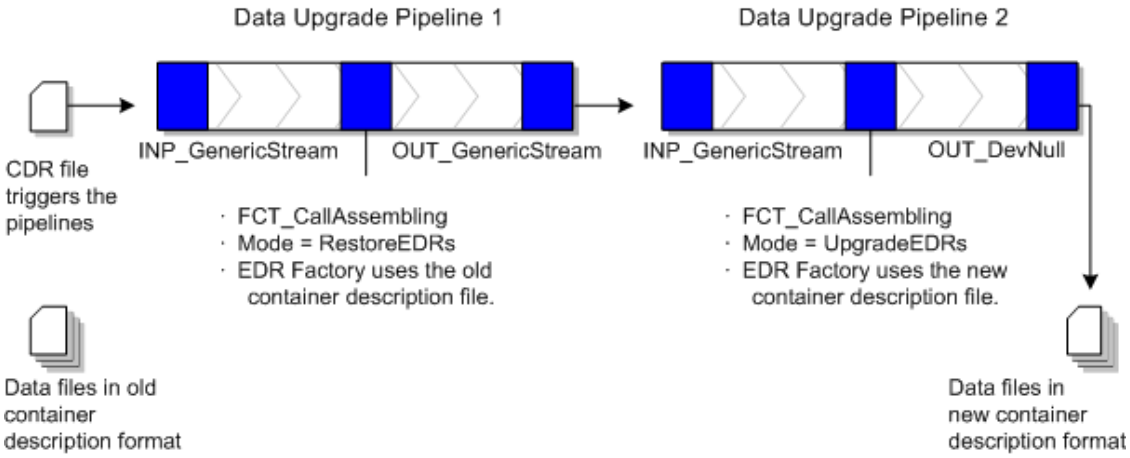
are partially assembled at the time you shut down a pipeline to make the EDR container description changes.

> **Note**
>
> The tasks in this section apply only to BRM implementations that use FCT_CallAssembling. If you use an external mechanism to assemble wireless records, the instructions in "Modifying and Loading the EDR Container Description" are all you need to modify your EDR container description.

To correctly rate the incomplete calls with your new EDR container description, you must first update the data file maintained by FCT_CallAssembling. The following procedure explains how to do this by running the EDRs through two miniature pipelines that you will create. After this conversion, you can process these calls through the pipeline set up to process call records using the new container description file. Before you run the pipelines, you will also create new input and output mapping and grammar files and a new stream file.

Figure 3-2 shows the process of upgrading stored EDRs to a new EDR container description.

Figure 3-2 Incomplete Call Description Update Process



> **Note**
>
> Using the **RestoreEDRs** and **UpgradeEDRs** modes, FCT_CallAssembling does not assemble calls. Instead, it only migrates them from one format to another.

Follow these steps to update EDRs being stored by FCT_CallAssembling to your new EDR format. This procedure assumes that you have already created your new container description file. For details see "Modifying EDR Container Descriptions".

Run this entire procedure for each EDR container being changed.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> If you are upgrading multiple pipelines to the new container description, you only need to create one set of data upgrade pipelines. You can use them to upgrade all FCT_CallAssembling pipelines getting the new container description.

1. Run the **pin_container_to_stream_format** utility on your old EDR container description file.

   For example:

   ```
   pin_container_to_stream_format -c oldcontainer.dsc -g OLD_ -m OLD_ -s OLD_
   ```

   This creates the stream, input and output grammar, and input and output mapping files that you will use to convert the partially assembled EDRs.
2. (Optional) Make any changes to the data fields in the new input grammar file to match your new EDR container description. The new container description file will have automatically supplied any necessary field *additions*. Edit the grammar file to specify any other changes to existing fields.
3. Create data upgrade pipeline 1. This pipeline will have only the following modules:
   - INP_GenericStream using your old input mapping file (actually any input mapping file can be used here).
   - FCT_CallAssembling. Set **Mode=RestoreEDRs** and set the **EdrFactory description** entry to your *old* container description file.
   - OUT_GenericStream using the output mapping created by **pin_container_to_stream_format** in Step 1.
4. Create data upgrade pipeline 2. This pipeline will have only the following modules:
   - INP_GenericStream using the input mapping file created by **pin_container_to_stream_format** in Step 1 (modified as needed).
   - FCT_CallAssembling. Set **Mode=UpGradeEDRs** and set the **EdrFactory description** entry to your *new* container description file created in Step 1.
   - OUT_DevNull (removes the single CDR file you send in to initiate the pipelines).
5. Use a semaphore to stop the rating pipeline that is receiving the EDR container change. Wait for it to stop completely.
6. Back up the following files used by FCT_CallAssembling in your rating pipeline:
   - The most recent **.dat** file
   - All **.EDR** files
   - The most recent **.INDEX** file
7. Copy the most recent **.dat** file used by FCT_CallAssembling in your rating pipeline to the data directory used by FCT_CallAssembling in data upgrade pipeline 1 and data upgrade pipeline 2.
8. Copy the **.EDR** file used by FCT_CallAssembling from your rating pipeline to the EDR directory used by FCT_CallAssembling in data upgrade pipeline 1.
9. Copy the most recent **.INDEX** file used by FCT_CallAssembling in your rating pipeline to the index directory used by FCT_CallAssembling in data upgrade pipeline 1.
10. Use a semaphore to start the data upgrade pipelines.

Copyright © 2025, 2026, Oracle and/or its affiliates.

11. Put one disposable CDR file in the input directory for data migration pipeline 1 to trigger the data upgrade. This file need only contain a single CDR, but it must use your old EDR container description format.

> **Note**
>
> This CDR is needed only to start EDR processing; it will get dropped during processing.

12. Ensure that data upgrade pipeline processes have finished. The time required for these pipelines to update all EDRs varies with the number EDRs being updated.
13. Use a semaphore to stop the data upgrade pipelines.
14. Move the **.dat** and **.EDR** files from the data upgrade pipeline 2 output directory to the input directory of the FCT_CallAssembling pipeline receiving the EDR container change.
15. Install the new EDR container description in your rating pipeline, and start the pipeline.

This completes the process of upgrading all EDRs stored in FCT_CallAssembling while upgrading your EDR container description. These incomplete calls are processed normally.

About Serviceless Accounts as Charge Sharing Owners

Charge sharing group owners can be accounts that have no services. This enables you to bypass purchasing specific services for corporate accounts that are used exclusively to pool employee charges. For example, you can create a corporate account that assumes telephone, email, and IP charges for all employees in the marketing group. But, you do not have to purchase GSM, email, or IP services for that corporate account. If you set up a serviceless account as a charge sharing group owner, BRM applies the member charges to that account's default balance group.

Pipeline Manager uses the create, modify, and delete sharing group events to determine the owner of the sharing group. If the owner is an account without services, the PIN_FLD_SERVICE_OBJ value is NULL. If the owner is a service, the PIN_FLD_SERVICE_OBJ value is the POID of the service.

To create a charge sharing owner that is a serviceless account, perform the following:

1. Enable Pipeline Manager to accept serviceless accounts by setting the DAT_AccountBatch **LoadAccountForSharingOnly** registry entry to **True**. This prevents the pipeline from rejecting serviceless accounts if they are owners of balance element sharing groups. By default, this entry is set to **False.**
2. Create a serviceless account. First create a package that has no bundles. Then, purchase the package when you create the account. When you create a charge sharing group for this account, the result is an account-level charge sharing group not associated with any specific services.

Mapping Subscription Services in the Pipeline Manager Database

To rate subscription services in a pipeline, the data that defines the subscription service object must be mapped in the Pipeline Manager database. Sample mapping is provided by BRM for some service objects. However, if you set up a service object for the subscription service that

Copyright © 2025, 2026, Oracle and/or its affiliates.

is not already configured in the Pipeline Manager database, you must map that service to the data that defines it.

Use Pricing Center or PCC to map the subscription service in the following Pipeline Manager database tables:

- IFW_SERVICE specifies the pipeline rating service code to BRM rating service code mapping; for example, map **GSM** to **/service/telco/gsm**.
- IFW_SERVICE_MAP specifies the external data used to determine the internal service code. You define the mapping when you set up service code mapping.
- IFW_REF_MAP specifies the service-type-to-event-type mapping so that services are associated with usage events. This database table also specifies the prefix for the data that identifies an account; for example, MSISDN or IMSI.

> **Note**
>
> Be sure to map a corresponding delayed event type to your subscription service object. For example, if you map to the usage event type **/event/session/telco/gsm**, also map to **/event/delayed/session/telco/gsm**. The delayed event type is used by Rated Event (RE) Loader when loading the rated events into the BRM database.
>
> If the delayed event type does not exist in the BRM database, you must create one. S

- IFW_ALIAS_MAP specifies which data to use for looking up an account in order to get the parameters for rating usage. You define the alias mapping when you set up EDR container fields.

  To define the alias mapping by using Pricing Center or PCC:

  1. From the **Pipeline Setup Toolbox**, select **EDR - EDR Container Description**.
  2. In the EDR Container Description dialog box, select the **ALL_RATE** sample container description and click **Edit**.
  3. Click the **Alias Map** tab and enter the alias data information for the subscription service.

Setting Up Batch Rating to Assign Items Based on Event Attributes

To set up Pipeline Manager to assign item tags to events based on event attributes:

1. Configure the following Pipeline Manager modules:
   - DAT_ItemAssign
   - FCT_ItemAssign
   - FCT_BillingRecord
2. Create a custom iScript that assigns item tags based on event attributes.
3. Load the rated events into the BRM database to update account balances and to create or update bill items.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Create a custom iScript that assigns item tags based on event attributes and fills in the DETAIL.ITEM_TAG field in the EDR container.

To enable your custom iScript to run in a pipeline, you must add an entry for it in the **wireless.reg** registry file. Configure this iScript to run *after* FCT_Account and *before* FCT_BillingRecord.

Sample registry entry:

```
# iScript to populate DETAIL.ITEM_TAG
#
IScript
{
  ModuleName = FCT_IScript
  Module
  {
   Active      = TRUE
   Source      = File
   Scripts
   {
    ItemTag
    {
     #iScript file that you created
     FileName = ./ISC_ItemTag.isc
    }
   }
  }
}
# end of iScript
```

Loading the Rated Events into the BRM Database

You use Rated Event (RE) Loader to load rated events into the BRM database. Before updating items in the database, RE Loader checks the updater flag in the RE Loader **Infranet.properties** file. If the flag value is **1**, RE Loader creates in the database the new item objects that were added. By default, the flag is set to **1**.

How Batch Rating Assigns Custom Bill Items

BRM batch rating uses the FCT_ItemAssign pipeline module to assign items based on event and service combinations and uses custom iScripts to assign items based on event attributes.

To assign items to events, Pipeline Manager performs these tasks:

1. During initialization, the DAT_ItemAssign module loads the specified **/config/item_types** object into memory and reserves a pool of POID IDs. If the information in the DAT_ItemAssign config object changes, you can use the DAT_itemAssign module's **Reload** semaphore to refresh the configuration changes.
2. Your custom iScript assigns an item tag based on event attributes to the DETAIL.ITEM_TAG EDR field.
3. FCT_ItemAssign calls DAT_ItemAssign with the item tag in the DETAIL.ITEM_TAG EDR field.
4. DAT_ItemAssign retrieves the item POID list from the DAT_AccountBatch module.
5. DAT_ItemAssign retrieves the item type for the given item tag from the **config/item_types** object in memory and searches the POID list for a matching item type.

Copyright © 2025, 2026, Oracle and/or its affiliates.

6. If DAT_ItemAssign finds a matching POID, it returns that item POID (for example, **1 /item/new_york m m**) to FCT_ItemAssign.

   If DAT_ItemAssign does not find a matching POID, it creates a new POID ID from the POID pool it reserved and returns the new POID ID to FCT_ItemAssign.

   When the DAT_ItemAssign module creates new items, it updates DAT_AccountBatch with the new items it created; for example, **1 /item/new_york m m**.

   If the DETAIL.ITEM_TAG field is NULL, DAT_ItemAssign returns a default item POID from the item POID list.

7. The FCT_ItemAssign module assigns the item POID that it retrieves to the DETAIL.CUST_A.PRODUCT.SERVICE_USED_ITEM_POID EDR field for the product used for rating the event.

8. For sponsored usage events, the following occurs:

   a. The FCT_BillingRecord module queries the DAT_ItemAssign module for items when required.

   b. The DAT_ItemAssign module returns the pre-created items of type **/item/sponsor** to FCT_BillingRecord for sponsored events.

   c. When the DAT_ItemAssign module creates new items, it updates DAT_AccountBatch with the new items it created; for example, **1 /item/sponsor m m**.

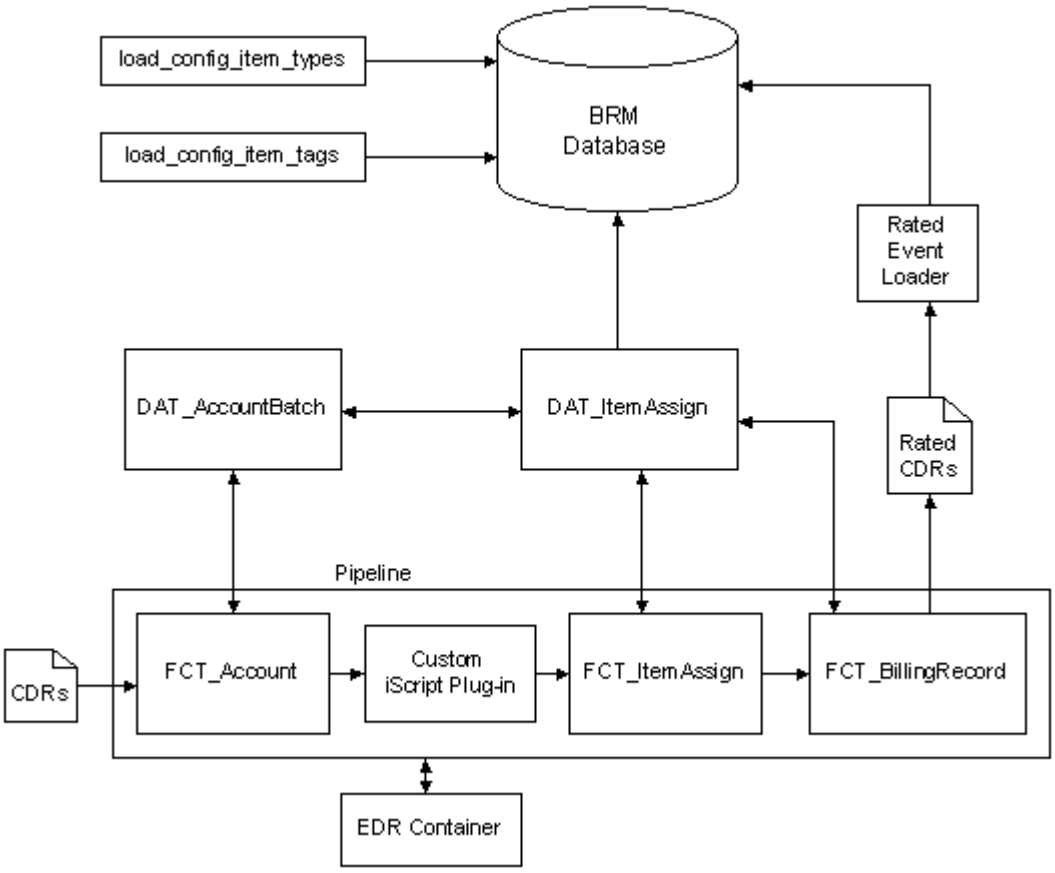
9. Pipeline Manager generates the rated events.

You use Rated Event (RE) Loader to load the rated events into the BRM database and to update the account balances and create or update bill items.

Figure 3-3 shows how items are assigned to events:

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 3-3 Assignment of Items to Events



How Batch Rating Assigns Custom Bill Items to Events for Balance Impacts

BRM batch rating uses the FCT_ItemAssign pipeline module to assign items based on event and service combinations and uses custom iScripts to assign items based on event attributes and balance impacts. The FCT_BillingRecord pipeline module converts impacts from charge offers (ChargePacket), discount offers (DiscountPacket), and taxes (TaxPacket) to balance impacts.

To assign custom bill items to events for balance impacts, Pipeline Manager performs the following tasks:

1. During initialization, the DAT_ItemAssign module loads the appropriate **/config/item_types** object into memory and reserves a pool of POID IDs. If the information in the DAT_ItemAssign config object changes, you can use the DAT_itemAssign module's **Reload** semaphore to refresh the configuration changes.
2. Your custom iScript does the following:
   a. Assigns an item tag based on event attributes to the DETAIL.ITEM_TAG EDR field.
   b. Changes the item POID for a specific balance impact in the event, using the following ChargePacket, DiscountPacket, and TaxPacket EDR tag container fields:
      - DETAIL.ASS.CBD.CP.ITEM_TAG
      - DETAIL.ASS.CBD.DP.ITEM_TAG
      - DETAIL.ASS.CBD.TP.ITEM_TAG

Copyright © 2025, 2026, Oracle and/or its affiliates.

3. If the ChargePacket, DiscountPacket, and TaxPacket EDR container tag field values are NULL, FCT_ItemAssign calls DAT_ItemAssign with the item tag from the DETAIL.ITEM_TAG EDR field.

   If the ChargePacket, DiscountPacket, and TaxPacket EDR container tag field values are not NULL, FCT_ItemAssign calls DAT_ItemAssign with the item tag from the ChargePacket, DiscountPacket, and TaxPacket EDR container fields and the item tag from the DETAIL.ITEM_TAG EDR field.

4. DAT_ItemAssign retrieves the item POID list from the DAT_AccountBatch module.
5. DAT_ItemAssign retrieves the item type for the given item tag from the **config/item_types** object in memory and searches the POID list for a matching item type.
6. If DAT_ItemAssign finds a matching POID, it returns that item POID (for example, **1 /item/new_york m m**) to FCT_ItemAssign.

   If DAT_ItemAssign does not find a matching POID, it creates a new POID ID from the POID pool it reserved and returns the new POID ID to FCT_ItemAssign.

   When DAT_ItemAssign creates new items, it updates DAT_AccountBatch with the new items it created; for example, **1 /item/new_york m m**.

   If the DETAIL.ITEM_TAG field is NULL, DAT_ItemAssign returns a default item POID from the item POID list.

7. The FCT_ItemAssign module does the following:

   a. Assigns the item POID that it retrieves to the DETAIL.CUST_A.PRODUCT.SERVICE_USED_ITEM_POID EDR field for the charge offer used for rating the event.

   b. Assigns the item POID that it retrieves from ChargePacket, DiscountPacket, and TaxPacket EDR container tag fields respectively to:

      - DETAIL.ASS.CBD.CP.ITEM_POID
      - DETAIL.ASS.CBD.DP.ITEM_POID
      - DETAIL.ASS.CBD.TP.ITEM_POID

   > **Note**
   >
   > If no item tags are configured for ChargePacket, DiscountPacket, and TaxPacket, FCT_ItemAssign replaces the corresponding packet's item POID with the updated POID from DETAIL.CUST_A.PRODUCT.SERVICE_USED_ITEM_POID.

8. For sponsored usage events, the following occurs if the SplitSponsorItemByMember business parameter is enabled:

   a. Pipeline Manager receives input from the **config_item_tags.xml** and **config_item_type.xml** files to determine the sponsor item type.

   b. Assigns a type-only POID; for example, **/item/sponsor/usage-1**.

   c. The RE Loader assigns an appropriate sponsor item instance to the sponsor balance impacts.

9. Pipeline Manager generates the rated events.

You use RE Loader to load the rated events into the BRM database and to update the account balances and create or update bill items.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Creating a Batch Rating iScript for Balance Impacts

BRM batch rating uses custom iScripts to assign items based on event attributes.

Create a custom iScript that does the following:

1. Assigns an item tag based on event attributes to the DETAIL.ITEM_TAG EDR field.
2. Changes the item POID for a specific balance impact in an event, using the following fields in the EDR container:
   - DETAIL.ASS_CBD.CP.ITEM_TAG
   - DETAIL.ASS_CBD.DP.ITEM_TAG
   - DETAIL.ASS_CBD.TP.ITEM_TAG

To enable your custom iScript to run in a pipeline, you must add an entry for it in the **wireless.reg** registry file. Configure this iScript to run *after* FCT_Apply_Balance and *before* FCT_ItemAssign.

Verifying Item-Tag-to-Item-Type Mapping

You can generate a log file that contains the item-tag-to-item-type mapping information from the DAT_ItemAssign memory.

To generate a log file of the mapping:

1. Create a semaphore registry file with following entry:

   ```
   ifw.DataPool.ItemAssignDataModule.Module.PrintData=TagTypeMap.txt
   ```

2. Copy the file into the semaphore directory. The default directory for semaphore files is *BRM_home*/**ifw/semaphore**.

Pipeline Manager generates the **TagTypeMap.txt** file, which contains the tag and type mapping from the DAT_ItemAssign module memory.

For example, the file contains entries as follows:

```
Total number of Tag and Type Mapping entries: 3
------------------------------------------------------------
Tag : Type
------------------------------------------------------------
cycle_forward : /item/cycle_forward
misc : /item/misc
newyork: /item/newyork
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

4

Configuring Resource Rounding

This chapter describes how to configure balance impact rounding rules if you are using Oracle Communications Billing and Revenue Management (BRM) Pricing Center. This data is used by both real-time rating and pipeline batch rating.

Topics in this document:

- About Resource Rounding
- Configuring Resource Rounding

> **Note**
>
> If you are using Oracle Communications Pricing Design Center (PDC) to create product offerings, see "Configuring Balance Impact Rounding" in *PDC Creating Product Offerings*.

About Resource Rounding

BRM enables you to create various rounding rules for different resources, events, and processes such as rating and discounting. You create rounding rules for several reasons:

- To increase the accuracy of rating and discounting results.
- To process usage fees more efficiently. For example, an infinite number such as 5.333... is more easily processed when it is rounded.
- To round for various currencies that use a different number of digits to the right of the decimal (for example, 10.25 dollars and 10 yen).
- To comply with currency conversion rules.
- To bill customers an amount that they can actually pay.

You configure rounding by editing the **pin_beid** file and loading the contents of the file into the ***/config/beid*** object in the BRM database. For more information, see "Configuring Resource Rounding".

About Rounding Rules

You can configure resource rounding based on the following rounding criteria:

- The *rounding scale* is the number of significant digits to the right of the decimal point. For example, a scale of 2 applied to 10.321111 rounds to 10.32.
- The *rounding mode* defines whether the number is rounded up, down, or not at all, based on the value of the digit following the last significant digit. See "About Rounding Modes".
- The process to which the rounding configuration applies. You can specify one of these processes: rating, discounting, taxation, and accounts receivable (A/R). A/R includes actions such as billing, payments, adjustments, cancellations, and G/L reporting.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Specifying the process enables you to round differently based on the operation. For example, you can round up using six decimal places for rating and round down using two decimal places for billing.

- The event type to which the rounding configuration applies. This enables you to round differently for events that represent specific types of usage, cycle fees, discounts, and rollovers. For example, US dollars and purchase events (**/event/purchase**).

Table 4-1 shows how a US dollars resource and purchase event combination can be rounded various ways for various processes.

Table 4-1 Currency Resource and Event Type

For a US Dollars Resource	For This Process	Use This Rounding Scale	Use This Rounding Mode
Event type = **/event/billing/product/fee/purchase**	Rating	6	Down
Event type = **/event/billing/product/fee/purchase**	Discounting	6	Up
Event type = **/event/billing/product/fee/purchase**	A/R	2	Nearest
Event type = **/event/billing/product/fee/purchase**	Taxation	2	Nearest

You can specify any combination of scale and mode for resource, event, and process combinations.

After each process that performs rounding (rating, discounting, and so forth), the balance impact of the event contains the rounded amount.

To configure rounding, you specify the rounding rules in the resource configuration file (*BRM_home***/sys/data/pricing/example/pin_beid**), and then you load the file by using the **load_pin_beid** utility. The **pin_beid** file contains default rounding rules that you can use. For more information, see "Configuring Resource Rounding".

> **Note**
>
> To round noncurrency aggregation counter balances for discounting purposes, use Pipeline Manager discount expressions. For more information, see "About Rounding Aggregation Counter Resources for Discounting".

How BRM Applies Rounding

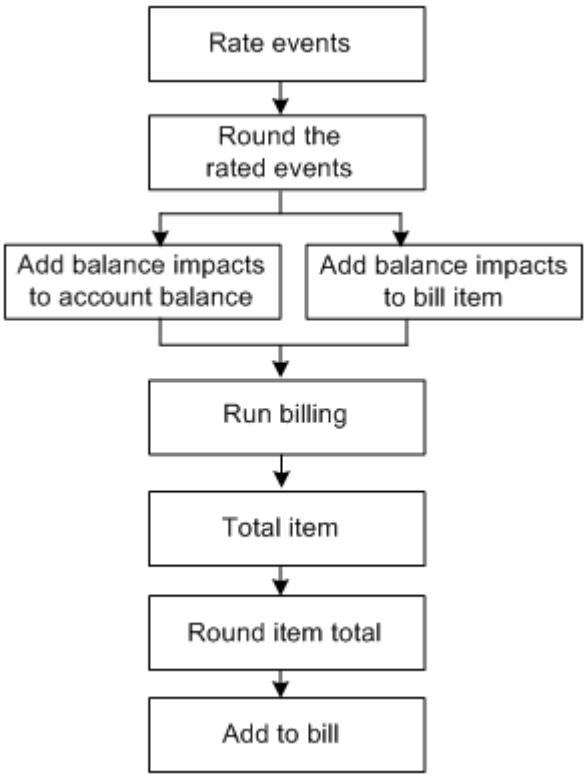
Rating, discounting, and taxation produce balance impacts, which are rounded and applied to the customer's account balance. The balance impacts are rounded to the scale and mode configured for the event, resource, and process combination.

Balance impacts for an account are stored in bill items, which are associated with the customer's bill. When you run billing, the item totals are rounded according to the A/R rounding rule. Item totals are rounded before being added to the bill so that the bill itself never needs rounding. Before billing, item totals are unrounded and their amounts reflect the scale configured for the associated events.

Figure 4-1 show the general process of balance impact rounding.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 4-1 BRM Balance Impact Rounding



Both real-time rating and pipeline batch rating perform rounding. During pipeline rating, the pipeline rounds the amount in each charge and discount packet.

To round in a pipeline, you configure the Pipeline Manager rounding module. See "Setting Up Rounding in Pipeline Manager". To configure the rounding rules, see "Configuring Resource Rounding".

Assuming that rounding is configured for all events and all processes (rating, discounting, taxation, and A/R), rounding is performed in the following order:

1. **Purchase and cycle fees**. When a plan is purchased, the purchase and cycle fees are rounded, if necessary, and a balance impact is generated with the rounded amount. Purchase and cycle fees typically only need rounding when a user purchases or cancels a plan in the middle of the billing cycle. In this case, cycle fees and noncurrency grant amounts are prorated, if necessary, and then rounded.
2. **Usage fees**. As customers use their services, the usage events are rated, the usage fees are rounded, and balance impacts are generated with the rounded amounts.
3. **Discount fees**. If a usage discount applies, the discount is calculated on the rounded usage fee, and then the discount is rounded and a discount balance impact is generated with the rounded amount. If there are multiple usage discounts, each discount is rounded in sequence.
4. **Taxes**. If tax should be applied to the event, the tax is calculated on the fee rounded from the previous stage, and then the tax is rounded and a balance impact is generated with the rounded amount.

Copyright © 2025, 2026, Oracle and/or its affiliates.

5. **Billing discounts**. When you run billing, if a billing-time discount applies, it is applied to the total of the rounded usage items, and then the discount is rounded and a discount balance impact is generated with the rounded amount.
6. **Bills**. When the bill is generated, the total amount in each item is rounded, and then all item totals are summed and included in the bill.
7. **Rollover**. If there are any rollover amounts, the rollover is calculated on the total rounded usage, and then the rollover amount is rounded and applied to the next billing cycle. For example, if 100 minutes can be rolled over and the customer used 60.4 minutes, the unused amount of 39.6 minutes is rounded and then added to the account balance for the next cycle.

   To round for rollover, you configure a rounding rule for the noncurrency resource and cycle forward event combination. Rollover is typically rounded up.

About Rounding and A/R Actions

When entering amounts for A/R actions such as adjustments and refunds, customer service representatives (CSRs) typically use *natural scale*: that is, the scale commonly used in the marketplace for that resource. For example, a person who purchases a book using US dollars cannot make change smaller than one cent (.01 dollars), making 2 the natural scale for US dollars. However, if a CSR reverses or adjusts an event before billing, the scale used is the one in the event's balance impact. By default, this is the natural scale, unless you change this to use a scale other than natural.

About Rounding Billed and Unbilled Items

The balance impacts of events associated with items can have a high scale. Before billing, item totals reflect the scale of their associated events. During billing, item totals are rounded using the A/R rounding configuration so that customers' bills display the natural scale for the resource. However, the events associated with billed items still retain their pre-billing scale.

If any operation is performed on billed items: for example, event-level and item-level adjustments, BRM rounds the balance impact of the operation according to the A/R rounding rule to maintain G/L integrity. Because actions on billed items are rounded when they occur, only pending items are rounded when billing is run.

About Rounding for Specific Event Types

When you configure a rounding rule for a specific event type, such as cycle or session events, that rounding rule applies to those events only for the process specified. If you do not configure a rule for every process, all other processes for that event type use the default rounding rule defined in the **pin_beid** file. The default rounding rule specifies natural scale for all events and the rounding mode most commonly used for the process. A rule for all events is specified by using an asterisk (*****) as the default event type.

For example, given the configuration in Table 4-2, during rating, session events are rounded down and have a scale of 6. During taxation, however, session events use the default rule of rounding to the nearest with a scale of 2.

Table 4-2 Rounding for Specific Event Types

Resource	Event Type	Process	Rounding Scale	Rounding Mode
US dollars	**/event/session**	Rating	6	Down
US dollars	***** (all events)	Taxation	2	Nearest

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Rounding Aggregation Counter Resources for Discounting

For discount rounding, the rounding configurations in the **pin_beid** file are used to round only the balance impacts of discounting, not input balances such as aggregation counters. To round aggregation counter balances in a pipeline, you use pipeline discount expressions when you configure your discounts. To configure aggregation counter rounding, see "Configuring Rounding Rules for Aggregation Counter Resources".

About G/L Report Rounding

G/L report rounding uses the rounding rule configured for A/R actions. The rounded totals in G/L reports might differ slightly from the total of the rounded bills. This is because item totals are rounded for billing, and journal entries are rounded for G/L reports. Also, G/L reports are rounded differently before and after billing. For more information, see "About Rounding and G/L Reports" in *BRM Collecting General Ledger Data*.

If there is any difference between the rounded journal entries in G/L reports and the rounded bill items, BRM records the difference in the G/L. You configure a G/L ID for the rounding difference and configure BRM to record the differences. See "Configuring to Record Rounding Differences in the G/L".

About Rounding Modes

A rounding mode defines whether a number is rounded up, down, or not at all. The rounding mode rounds to the specified scale. For example, using a scale of 2, rounding up 10.2369 results in 10.24. If the scale is 3, rounding up results in 10.237.

The following rounding mode values that you can use in the **pin_beid** file are defined in the *BRM_home***/include/pin_bill.h** file:

- **0**: PIN_BEID_ROUND_NEAREST

  This mode rounds up or down depending on the value of the digit following the last significant digit. If the additional digit is 0-4, the last significant digit remains the same. If the additional digit is 5-9, the last significant digit is rounded up. For example, if the scale is 2, 10.144 rounds to 10.14 and 10.145 rounds to 10.15. This is the most common rounding method.

- **1**: PIN_BEID_ROUND_UP

  This mode rounds up when the digit following the last significant digit is greater than 0. For example, If the scale is 2, 10.151 rounds to 10.16. If the scale is 1, it rounds to 10.2.

- **2**: PIN_BEID_ROUND_DOWN

  This mode truncates all digits following the last significant digit. For example, if the scale is 2, 10.159 rounds to 10.15. If the scale is 1, it rounds to 10.1.

- **3**: PIN_BEID_ROUND_EVEN

  This mode rounds one of three ways depending on the value of the digit following the last significant digit:

  - If it is less than 5, truncate all digits following the last significant digit.
  - If it is greater than 5, round up.
  - If it is 5, round to the nearest even digit. For example, if the scale is 2, 10.155 and 10.165 both round to 10.16 because 6 is an even number.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- **4**: PIN_BEID_ROUND_FLOOR

  This mode rounds numbers toward a negative value (that is, the rounded number is always less than the unrounded number). This enables you to round balance impacts so that customers always benefit. For example, if the scale is 2, a credit to a customer of -7.999 is rounded to -8.00, and a debit of 7.999 is rounded to 7.99.

- The following two modes perform the same rounding as their non-alternative counterparts (ROUND_FLOOR and ROUND_DOWN), except that they compensate for possible loss of precision when rounding down by first rounding with a mode of NEAREST using a scale that is two digits greater than the scale you configure.
  - **5**: PIN_BEID_ROUND_FLOOR_ALT
  - **6**: PIN_BEID_ROUND_DOWN_ALT

  For more information, see "About Rounding Modes That Correct for Loss of Precision".

About Rounding Modes That Correct for Loss of Precision

Some calculations produce results that are slightly less than expected when a value is rounded down. For example, when BRM prorates a $60.00 cycle fee for 20 out of 30 active days, the calculation is (20/30) * $60.00. The expected result is a fee of $40.00. However, because 20/30 evaluates to 0.666..., when this is multiplied by 60 and rounded down, the actual result is a fee of $39.99.

BRM provides two alternative rounding modes that compensate for possible precision loss when rounding down: ROUND_DOWN_ALT and ROUND_FLOOR_ALT. These modes perform the same rounding as their non-alternative counterparts (ROUND_DOWN and ROUND_FLOOR) after first compensating for loss of precision.

> **Note**
>
> ROUND_DOWN_ALT and ROUND_FLOOR_ALT are not supported in the BRM PCM Java API and in discount expressions.

When these modes are used, if a decimal should be rounded down, BRM performs two rounding functions: The decimal is rounded by using the ROUND_NEAREST rounding mode and a scale that is two more than the scale that you request. It is then rounded down.

For example, if you configure the rounding mode as ROUND_DOWN_ALT and a rounding scale of 2, and the decimal to round is 7.999..., BRM truncates the infinite decimal to the system maximum and then rounds this decimal to the nearest using a scale of 4 (2 more than the configured scale of 2), which results in 8.0000. This decimal is then rounded down using the configured scale of 2, resulting in 8.00 as shown in Table 4-3:

Table 4-3 Rounding Modes That Correct for Precision

Do This	Why
7.999...	Original decimal
7.99999999999999	Truncated to the system max
8.0000	ROUND_NEAREST, using requested scale + 2
8.00	ROUND_DOWN to the requested scale of 2

Copyright © 2025, 2026, Oracle and/or its affiliates.

Had the original decimal of 7.999... not been rounded to the nearest first and only rounded down, the result would be 7.99.

To compensate for possible loss of precision, the alternative rounding modes consider two decimal places more than the non-alternative rounding modes. Therefore, the greatest amount that will be modified by using the alternative rounding modes, and still compensate for loss of precision, is less than the greatest amount that will be modified by using the non-alternative rounding modes.

When Rounding Is Not Applied

When the requested rounding scale is greater than the scale of the number being processed, rounding is not required. This occurs when:

- You configure a rounding scale equal to or greater than the maximum number of digits allowed by the system. For example, if the system allows 15 digits and you set the rounding scale to 15 or greater, rounding has no effect.
- You call a rounding function and request a scale that is equal to or greater than the current scale of the decimal: for example, when the decimal is 10.89766 and the scale requested is 5 or greater.
- A computation or expression results in a decimal with a scale that is equal to or less than the configured or requested scale: for example, when a computation results in the decimal 1.98 and the configured scale is 2 or greater.

Configuring Resource Rounding

To set up rounding, perform the following tasks:

- Configuring Rounding Rules
- Configuring Rounding Rules for Aggregation Counter Resources
- Configuring to Record Rounding Differences in the G/L
- Setting Up Rounding in Pipeline Manager

About Configuring Rounding Rules

You define rounding rules in the balance element ID (BEID) configuration file (B*RM_Home*/**sys/data/pricing/example/pin_beid**) and then run the **load_pin_beid** utility to load the contents of the file into the **/config/beid** object in the BRM database. See "Configuring Rounding Rules".

The **pin_beid** file contains default rounding rules based on the most commonly used rounding for the resource and operation. The default rules specify a configuration for all events by using an asterisk (*****) as the default event type. For an explanation of the configuration syntax and fields, see the **pin_beid** file.

You configure rounding for resource and event type combinations (for example, US dollars and **/event/session/telco/gsm** events). For each resource and event combination, you specify a rounding scale, a rounding mode, and the process for which this rule applies. For the process, you specify one of these enumerated values in the **pin_beid** file:

- Rating = **0**
- Discount = **1**
- Taxation = **2**

Copyright © 2025, 2026, Oracle and/or its affiliates.

- A/R = **3**

To configure an event and resource rule for every process, you add an entry for each process.

> **Note**
>
> - You do not need to configure rounding for all of the processes. However, you must configure rounding for every resource and event type that has a balance impact.
> - If you add resources that are not already in the **pin_beid** file, you should always include a default rounding configuration for that resource that applies to all event types. Any event type that is not explicitly specified for the resource uses this default rounding rule.
> - You can use a single asterisk (*****) to denote a default value such as any event type. However, in combination with other characters, you must use valid regular expressions. For example, specifying **/event/*** is incorrect; specifying **/event/(.)*** is correct.
> - You can use regular expressions in the rounding configurations. For example, **/event/session/(.)*** matches all session events.

For more information, see "Configuring Rounding Rules".

Prioritizing Rounding Rules

BRM applies the first matching rounding rule in the **/config/beid** file. Therefore, if you have more than one rule that matches an event, resource, and process combination, add them in order of priority in the **pin_beid** file. For example, a configuration that matches a specific event type should be added before a configuration that matches all event types.

If there is no rounding rule in the **/config/beid** object that matches the resource, event, and process combination, and no default rule that applies to all events, the event is not rounded.

About Rounding Mode Values

The rounding modes you specify in the **pin_beid** file have corresponding modes in the following API components. However, their values are not all the same:

- **The BRM decimal data type functions**. When BRM invokes a decimal data type function, it converts the rounding mode in the **/config/beid** object into the corresponding rounding parameter used by the decimal data type function. If you call decimal data type functions in custom applications, specify the function's rounding mode, not the BEID rounding mode.
- **The BAS Decimal class**. The FCT_Rounding pipeline module converts the rounding mode in the **/config/beid** object into the corresponding rounding mode used by the BAS rounding method in Pipeline Manager. If you use the BAS Decimal rounding method in custom pipeline modules and iScripts, you should include the FCT_Rounding module in your pipeline. Otherwise, you will need to convert the rounding mode in the **/config/beid** object into the BAS rounding mode before calling the BAS rounding method.

Table 4-4 shows the BEID rounding modes and the corresponding rounding parameters for the decimal data type functions and the BAS rounding modes:

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 4-4 BEID Rounding Modes

BEID Rounding Mode (specified in pin_bill.h)	Rounding Mode Parameter for pbo_decimal functions	BAS Decimal Rounding Mode
0: PIN_BEID_ROUND_NEAREST	5: ROUND_HALF_UP	0: PLAIN
1: PIN_BEID_ROUND_UP	1: ROUND_UP	1: UP
2: PIN_BEID_ROUND_DOWN	2: ROUND_DOWN	2: DOWN and TRUNCATE
3: PIN_BEID_ROUND_EVEN	7: ROUND_HALF_EVEN	3: BANKERS
4: PIN_BEID_ROUND_FLOOR	4: ROUND_FLOOR	101: FLOOR
5: PIN_BEID_ROUND_FLOOR_ALT	8: ROUND_FLOOR_ALT	103: FLOOR_ALT
6:PIN_BEID_ROUND_DOWN_ALT	9: ROUND_DOWN_ALT	102: DOWN_ALT
N/A	3: ROUND_CEILING	N/A
N/A	6: ROUND_HALF_DOWN	N/A
N/A	10: ROUND_UNNECESSARY	N/A

Configuring Rounding Rules

To configure rounding rules, you edit the BEID configuration file and then run the **load_pin_beid** utility to load the contents of the file into the **/config/beid** object in the BRM database.

> **Note**
>
> The **load_pin_beid** utility needs a configuration (**pin.conf**) file in the directory from which you run the utility.

To configure resource rounding:

1. Edit the **pin_beid** file in *BRM_home***/sys/data/pricing/example**. The **pin_beid** file includes instructions.

   > **Note**
   >
   > The **load_pin_beid** utility overwrites the existing resources. If you are updating resources, you cannot load new resources only. You must load complete sets of resources each time you run the **load_pin_beid** utility.

2. Save the **pin_beid** file.
3. Use the following command to run the **load_pin_beid** utility:

   ```
   load_pin_beid pin_beid
   ```

   If you do not run the utility from the directory in which the file is located, you must include the complete path to the file. For example:

   ```
   load_pin_beid BRM_home/sys/data/pricing/example/pin_beid
   ```


Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> If you copy the **pin_beid** file to the directory from which you run the **load_pin_beid** utility, you do not have to specify the path or file name. The file must be named **pin_beid**.

For more information, see "load_pin_beid".

4. Stop and restart the Connection Manager (CM).

To verify that the network elements were loaded, you can display the **/config/beid** object by using Object Browser, or use the **robj** command with the **testnap** utility.

After loading the **pin_beid** file, functions that calculate fees and discounts use the rounding rules in the **/config/beid** object.

Configuring Rounding Rules for Aggregation Counter Resources

You use Pricing Center to configure rounding rules for aggregation counter resources. When you set up your discount configurations, use discount expressions to specify how to round the resource. Use the following discount expression syntax:

```
round(expression, rounding_scale, rounding_mode)
```

where expression defines the resource balance to round. This can be any discount expression. To round an aggregation balance, use the **Bal** expression. For more information, see the discussion about setting up discounts in the Pricing Center Help.

For example, to round a balance down to two decimal places for an aggregation counter resource with ID 100099, use the following expression:

```
round( Bal(1000099), 2, ROUND_DOWN )
```

About Rounding Modes for Discount Expressions

Rounding modes for discount expressions are equivalent to those you specify in the **pin_beid** file, but they have slightly different names. Table 4-5 lists the rounding mode values you can specify in discount expressions and their **pin_beid** file counterpart.

Table 4-5 Rounding Modes for Discount Expressions

Discount Expression Rounding Mode	Rounding Mode Used in the pin_beid File
ROUND_PLAIN	Nearest
ROUND_UP	Up
ROUND_DOWN	Down
ROUND_BANKERS	Even

For a definition of what these modes represent, see "About Rounding Modes".

Configuring to Record Rounding Differences in the G/L

To record any difference between rounded bill items and the rounded total in the G/L, perform the following tasks:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Defining a G/L ID for Rounding Differences
- Mapping the Rounding G/L ID to an Event
- Configuring BRM to Record Rounding Differences

For information about how rounding is performed in G/L reports, see the discussion about rounding and G/L reports in *BRM Collecting General Ledger Data*.

Defining a G/L ID for Rounding Differences

You define a G/L ID for rounding to include the rounding difference in G/L reports so that they can be accurately reconciled.

To define G/L IDs, you edit the G/L ID configuration file and then run the **load_pin_glid** utility to load the contents of the file into the **/config/glid** object in the BRM database.

> **Note**
>
> The **load_pin_glid** utility needs a configuration (**pin.conf**) file in the directory from which you run the utility.

To define a rounding G/L ID:

1. If necessary, edit the G/L ID configuration file in *BRM_home***/sys/data/pricing/example/pin_glid**. If the following entry is not present, add it:

```
#====================================================================
# G/L ID for rounding adjustments
#====================================================================
glid
id 1512
descr Rounding Epsilon
gl_acct billed         gross    rounding.debit    rounding.credit
gl_acct billed         net      rounding.debit    rounding.credit
gl_acct billed         disc     rounding.credit   rounding.debit
gl_acct billed_earned  gross    rounding.debit    rounding.credit
gl_acct billed_earned  net      rounding.debit    rounding.credit
gl_acct billed_earned  disc     rounding.credit   rounding.debit
gl_acct unbilled       gross    rounding.debit    rounding.credit
gl_acct unbilled       net      rounding.debit    rounding.credit
gl_acct unbilled       disc     rounding.credit   rounding.debit
gl_acct unbilled_earned gross   rounding.debit    rounding.credit
gl_acct unbilled_earned net     rounding.debit    rounding.credit
gl_acct unbilled_earned disc    rounding.credit   rounding.debit
)
```

> **Note**
>
> The **load_pin_glid** utility overwrites existing G/L IDs. If you are updating G/L IDs, you cannot load new G/L IDs only. You must load complete sets of G/L IDs each time you run the **load_pin_glid** utility.

2. Save and close the file.
3. Use the following command to run the **load_pin_glid** utility:

Copyright © 2025, 2026, Oracle and/or its affiliates.

```
load_pin_glid pin_glid_file
```

For more information, see the discussion about loading general ledger configuration data in *BRM Collecting General Ledger Data*.

Mapping the Rounding G/L ID to an Event

Because the rounding difference is not a rated event, you must map the G/L ID to an event type. G/L ID mapping is defined in the **reasons.***locale* file. You can find a sample of this file in the *BRM_home***/sys/msgs/reasoncodes** directory. The sample file is named **reasons.en_US** and contains the following default entry for the rounding G/L ID mapping:

```
DOMAIN = "Others" ;
STR
     EVENT-GLID
          . . .
          /event/journal/epsilon"        1512 ;
     EVENT-GLID-END
```

> **Note**
>
> **/event/journal/epsilon** is a dummy event type used for reference only.

To change the G/L ID for rounding, you must edit and reload the file. The G/L ID you define in the **reasons.***locale* and **pin_glid** files must match. See "Configuring to Record Rounding Differences in the G/L".

To map the G/L ID for rounding to an event, you use the **load_localized_strings** utility to load the contents of the file into the **/config/map_glid** object. When you run the **load_localized_strings** utility, use this command:

```
load_localized_strings reasons.locale
```

> **Note**
>
> - If you are loading a localized version of this file, use the correct file extension for your locale.
> - The **load_localized_strings** utility overwrites the existing G/L ID maps. If you are updating this object, you cannot load new G/L ID maps only. You must load complete sets of G/L ID maps each time you run the **load_localized_strings** utility.

For information on loading the **reasons.***locale* file, see the discussion about creating a localized version of BRM in *BRM Developer's Guide*.

Configuring BRM to Record Rounding Differences

By default, rounding differences are not recorded in G/L reports. You can enable this feature by modifying a field in the **billing** instance of the **/config/business_params** object.

You modify the **/config/business_params** object using the **pin_bus_params** utility.

To enable BRM to record rounding differences:

Copyright © 2025, 2026, Oracle and/or its affiliates.

1. Use the following command to create an editable XML file from the **billing** instance of the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsBilling bus_params_billing.xml
   ```

   This command creates the XML file named **bus_params_billing.xml.out** in your working directory. If you do not want this file in your working directory, specify the full path as part of the file name.

2. Search the XML file for the following line:

   ```
   <GenerateJournalEpsilon>disabled</GenerateJournalEpsilon>
   ```

3. Change **disabled** to **enabled**.

   > **Note**
   >
   > BRM uses the XML in this file to overwrite the existing **billing** instance of the **/config/business_params** object. If you delete or modify any other parameters in the file, these changes affect the associated aspects of the BRM billing configuration.

4. Save and close the file.

5. Use the following command to load the change into the **/config/business_params** object:

   ```
   pin_bus_params bus_params_billing.xml
   ```

   You should run this command from the *BRM_home*/**sys/data/config** directory, which includes support files used by the utility.

6. Read the object with the **testnap** utility or Object Browser to verify that all fields are correct.

7. Stop and restart the Connection Manager (CM).

Setting Up Rounding in Pipeline Manager

Pipeline Manager uses the rounding rules specified in the **/config/beid** object when applying rounding. To round balance impacts in Pipeline Manager, you configure the FCT_Rounding module. See "About Configuring the FCT_Rounding Module".

You can also set up separate rounding rules when defining pricing. See "About Rounding Rules in Pricing".

Rounding rules can conflict. For more information, see "Avoiding Rounding Conflicts in Pipeline Manager".

About Configuring the FCT_Rounding Module

The FCT_Rounding module finds the rounding rule in the **/config/beid** object based on the event, balance element, and process (rating, discounting, or taxation) combination and then rounds the amount in the relevant EDR packet.

Add FCT_Rounding to the pipeline after the processing module for which it is rounding:

- **To round for rating**, add this module after the FCT_RateAdjust module.
- **To round for taxation**, add this module after the ISC_TaxCalc iScript module.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- **To round for real-time discounting**, add this module after the FCT_Discount module.
- **To round for batch discounting**, add this module after the FCT_Discount module and before the FCT_ApplyBalance module.

You specify the process for which this module is rounding in the **Mode** entry of the module registry:

- **Rating** = Round the balance impact of rating.
- **Taxation** = Round the balance impact of taxation.
- **Discounting** = Round the balance impact of discounting.

For more information, see "Avoiding Rounding Conflicts in Pipeline Manager".

About Rounding Rules in Pricing

You specify one of these rounding methods when defining pricing:

- **None**: Only the rounding rules configured in FCT_Rounding are used.
- **Plain**: Round up if the last significant digit is 5 or greater.
- **Up**: Always round up to the next highest digit.
- **Down**: Always round down to the next lowest digit.
- **Bank**: If the last significant digit is 5, make it even.

If you specify a rounding method other than **None** for pricing, Pipeline Manager will use that rounding method during rating.

For more information, see "Avoiding Rounding Conflicts in Pipeline Manager".

Avoiding Rounding Conflicts in Pipeline Manager

For Pipeline Manager, it is possible to specify rounding rules in two places that apply to the same event. Pipeline Manager uses different rounding rules at different points in the rating process:

- The rounding method specified for the pricing is used by the rating module.
- The rounding method specified for the balance element and event is used in the FCT_Rounding module, which is usually run after the FCT_RateAdjust and FCT_Discount modules in the pipeline.

Because rounding can take place at these different points in the pipeline, you can get unexpected results. To avoid conflicts between different rounding rules, you can do one of the following:

- Choose **None** for the rounding method in the pricing. If you do this, only the rounding rules configured in FCT_Rounding are used.
- If you select a rounding method other than **None** in the pricing, ensure that the selected method is consistent with the rounding method that FCT_Rounding will use.

Rounding Examples

Rounding is performed after rating, discounting, taxation, and A/R actions such as billing and adjustments.

Table 4-6 shows the resulting balance impacts of rounded charges for rating, discounting, taxation, and billing. This example rounds to the nearest mode and uses these scales:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- 2 for purchase events, taxation, and A/R
- 5 for rating and discounting

Table 4-6 Rounding Examples

Action / Process	Calculated Charge	Rounded Charge & Balance Impact	Account Balance
Purchase plan Rate cycle fee	9.95	9.95	9.95
Use service Rate usage	5.23456789	5.23457	15.18456
Discount usage fee 10%	0.523456789	0.52346	14.66111
Tax usage event 3% (after discount)	0.1413333 = 3% * (rounded usage fee - rounded discount)	0.14	14.80111
Run billing Apply billing-time discount of 5% off total usage	0.24250... = 5% * usage item total after A/R rounding (The usage total does not include the cycle fee.)	0.2425	14.55861
Create bill	14.55861 = Total of all items	14.56 (No balance impact)	14.56

Correcting for Precision Loss When Rounding Down

This example shows the results of rounding when you use the ROUND_DOWN_ALT and ROUND_FLOOR_ALT modes. For more information, see "About Rounding Modes That Correct for Loss of Precision".

The ROUND_DOWN_ALT and ROUND_FLOOR_ALT modes produce different results than ROUND_DOWN and ROUND_FLOOR only when the three digits following the last significant digit are 995 or greater. (The last significant digit is the digit in the decimal place corresponding to the scale: If the scale is 2, the last significant digit in the number 1.23456 is 3.)

For example:

Table 4-7 shows some rounding results of the ROUND_DOWN_ALT and ROUND_FLOOR_ALT modes as compared to their non-alternative rounding counterparts (ROUND_DOWN and ROUND_FLOOR) for various decimal values and rounding scales.

Table 4-7 Rounding Results

Decimal	Scale	Rounding Mode	Rounding Mode	Rounding Mode	Rounding Mode
NA	**NA**	**DOWN**	**DOWN_ALT**	**FLOOR**	**FLOOR_ALT**
1.5256	2	1.52	1.52	1.52	1.52
-1.5256	0	-1	-1	-2	-2
12.8999...	0	12	12	12	12

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 4-7 (Cont.) Rounding Results

Decimal	Scale	Rounding Mode	Rounding Mode	Rounding Mode	Rounding Mode
NA	NA	DOWN	DOWN_ALT	FLOOR	FLOOR_ALT
12.8999...	1	12.8	12.9	12.8	12.9
12.8999...	2	12.89	12.90	12.89	12.90
-12.8999...	1	-12.8	-12.9	-12.8	-12.9
-12.8999...	2	-12.89	-12.900	-12.89	-12.90
-6.9990	2	-6.99	-6.99	-7.00	-7.00
-6.9990	3	-6.999	-6.999	-6.999	-6.999
7.999...	0	7	8	7	8
7.999...	1	7.9	8.0	7.9	8.0
7.999...	2	7.99	8.00	7.99	8.00
-7.999...	0	-7	-8	-8	-8
-7.999...	2	-7.99	-8.00	-8.00	-8.00

Rounding Using Different Modes

The aggregated effects of rounding on the final balance impact is determined by the mode and scale that you configure. The higher the scale, the less effect the rounding mode has on the final balance impact.

For example, Table 4-8 shows the impact of various rounding mode combinations for rating a usage fee of $1.1234567 that includes a 10% discount. Both rating and discounting use a scale of 6:

Table 4-8 Rounding Modes

Processing Stage (Rating and Discounting)	Rating: Round Down Discounting: Round Down	Rating: Round Down Discounting: Round Up	Rating: Round Up Discounting: Round Down	Rating: Round Up Discounting: Round Up
Round for rating	1.123456	1.123456	1.123457	1.123457
Calculate 10% discount	.1123456	.1123456	.1123457	.1123457
Round discounted amount	.112345	.112346	.112345	.112346
Apply discount to usage fee to get final balance impact	1.011111	1.011110	1.011112	1.011111

In this example, the difference in the final balance impacts is small because the scale is high and probably will not change the final amount on the bill. However, when many events are summed in an item, or the scale is small, such as 2 or 3, the differences become greater.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> If you calculate the discount without rounding the event, the configuration where rating is rounded down and discounting is rounded up returns the most accurate result. Therefore, this is the best mode configuration to use when you discount events.

Modifying a Rounding Rule

To modify a rounding rule, you must change the rule in the **pin_beid** file and reload the file by running the **load_pin_beid** utility. See "Configuring Resource Rounding".

Copyright © 2025, 2026, Oracle and/or its affiliates.

5

Managing Sub-Balances

This chapter describes how to manage sub-balances in Oracle Communications Billing and Revenue Management (BRM) Pricing Center. It describes balance impacts, credit limits, and rollovers.

Topics in this document:

- About Sub-Balances
- About Noncurrency Sub-Balances
- How Resources in Validity-Based Sub-Balances Are Updated
- About Configuring Sub-Balances
- Configuring Sub-Balances
- Sub-Balance Configuration Example
- Specifying the Order in Which Resource Sub-Balances Are Consumed
- About Rollovers
- About Rolling Over Free Resources during Plan Transition

About Sub-Balances

Each balance in a balance group can include one or more sub-balances. A balance includes sub-balances when portions of the balance are valid at different times or when a portion of the balance is a loan. For example, a balance of minutes might include 300 minutes that are valid only for the current month and 1000 minutes that never expire.

A currency sub-balance can store the balances for multiple services. For example, an account that owns two products that cost $25.00 per product has a starting currency sub-balance of $50.00, providing the services are associated with the same balance group and have the same validity period.

A currency balance can also store sub-balances for loans. For example, if an account subscribed to a monthly package that costs $25.00 only has a balance of $15.00 at billing time, you can grant a loan for the remaining $10.00. The balance would then consist of two sub-balances; a $15 regular sub-balance and a $10 loan sub-balance.

Sub-balances include the following information:

- The sub-type. This indicates whether the sub-balance is a loan or not.
- The start time and end time for which the sub-balance is valid.
  Common resources with the same validity periods are stored in the same sub-balance. Resources with unique validity periods are stored in separate sub-balances. For more information, see "How Resources in Validity-Based Sub-Balances Are Updated".

  For information about rounding the start times of cycle and purchase grants to midnight, see "Configuring Time-Stamp Rounding for Validity Period Start Times".
- The current amount of the sub-balance.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Consumed reservation (or active reserved amount) which tracks the consumed reservation and is used to set up notifications to the network.
  See the discussion on calculating reservation balances in *BRM Telco Integration*.
- The fields in the event record or object (referred to as “contributors") that contribute to how sub-balances are created, updated, and retrieved. For example, to retrieve the total available balance for a specific service, the service object is specified. To deduct free minutes for a phone call, the session object is specified. A separate sub-balance is kept for each unique contributor. See "About Sub-Balance Contributors".
- Rollover data such as the rollover period and the resource amount that is rolled over, if any. For more information, see "About Rollovers".
- The ID of the product or discount that granted the resource (referred to as “grantor object.")

You can configure sub-balances to track various types of resources and usage. See "About Configuring Sub-Balances".

About Noncurrency Sub-Balances

A noncurrency sub-balance typically has a limited validity period (for example, the period during which free minutes can be used). Noncurrency sub-balances can contain various types of resources, such as:

- Free minutes.
- Frequent flyer miles.
- Loyalty points.
- Number of emails or text messages.

A noncurrency sub-balance can also keep track of the total resources used for discounts that are shared among several accounts. In this case, the sub-balance acts as a counter to keep track of the total consumed resource.

When granting a noncurrency resource, if a sub-balance already exists, BRM compares the new balance data with the following data in the existing valid sub-balances:

- Contributor
- Grantor object
- Rollover data
- Valid-from date
- Valid-to date

If the data matches, BRM adds the amount to the existing sub-balance; otherwise, it creates a new sub-balance.

For information about configuring sub-balances, see "About Configuring Sub-Balances".

How Resources in Validity-Based Sub-Balances Are Updated

By default, BRM stores common resources with the same validity periods in the same sub-balance, provided they are associated with the same balance group. (You can create separate balance groups per service in Pricing Center. See "Tracking Resources by Service"). BRM automatically creates a new sub-balance for resources with a unique validity period, if one does not already exist.

Copyright © 2025, 2026, Oracle and/or its affiliates.

For example, an account owns two services that each include 100 free minutes that are always valid. The account has a balance of 200 free minutes stored in a single sub-balance. When the customer uses free minutes from each service, the free minutes are consumed from the common sub-balance.

Common resources with different validity periods are tracked in separate sub-balances. For example, an account owns two services that each include 100 free minutes. Free minutes for service 1 expire at the end of the month, and free minutes for service 2 expire at the end of the year. Each set of free minutes is stored in a separate sub-balance.

> **Note**
>
> When common, noncurrency resources are configured to start on first usage, BRM creates a new sub-balance for each resource whether or not they have the same validity period. See "About Noncurrency Sub-Balances That Start on First Usage".

You can specify the order in which sub-balances are consumed by setting up resource consumption rules. See "Specifying the Order in Which Resource Sub-Balances Are Consumed".

You can also limit how validity-based resources, such as free minutes, are consumed by configuring sub-balances. For example, you might want to limit usage of free minutes to a specific service or a specific call session. See "About Configuring Sub-Balances".

Configuring Time-Stamp Rounding for Validity Period Start Times

You can configure BRM to round time stamps to midnight for resources granted by cycle and purchase events.

See the following:

- Configuring Time-Stamp Rounding for Cycle Grants
- Configuring Time-Stamp Rounding for Purchase Grants

> **Note**
>
> When the resource's purchase, cycle, and usage start and end units are set to 1 (seconds), 2 (minutes), or 3 (hours), and the validity period is less than 24 hours, time stamps are not rounded, regardless of your system configuration. If the validity is greater than 24 hours, the cycle, usage, and purchase end time stamps are rounded for the purpose of calculating the scale to determine the cycle fee amount to charge.

Configuring Time-Stamp Rounding for Cycle Grants

To configure BRM to round time stamps to midnight for the resources granted by cycle events:

1. Go to *BRM_home*/**sys/data/config**.
2. Use the following command to create an editable XML file from the subscription instance of the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsSubscription bus_params_subscription.xml
   ```


Copyright © 2025, 2026, Oracle and/or its affiliates.

This command creates an XML file named **bus_params_subscription.xml.out** in your current directory. If you do not want this file in your current directory, specify the path as part of the file name.

3. In **bus_params_subscription.xml.out**, set **TimestampRounding** to **enabled**:

```
<TimestampRounding>enabled</TimestampRounding>
```

> ⚠ **Caution**
>
> BRM uses the XML in this file to overwrite the existing instance of the **/config/business_params** object. If you delete or modify any other parameters in the file, these changes affect the associated aspects of the BRM configuration.

4. Save and exit the file.
5. Rename the **bus_params_subscription.xml.out** file to **bus_params_subscription.xml**.
6. Use the following command to load your changes into the **/config/business_params** object:

```
pin_bus_params bus_params_subscription.xml
```

   You should run this command from the *BRM_home*/**sys/data/config** directory, which includes support files used by the utility. To run it from a different directory, see "pin_bus_params" in *BRM Developer's Guide*.

7. Read the object with the **testnap** utility or the Object Browser to verify that all fields are correct.

   For general instructions on using **testnap**, see "Using the testnap Utility to Test BRM" in *BRM Developer's Guide*. For information on how to use Object Browser, see "Reading Objects" in *BRM Developer's Guide*.

8. Stop and restart the Connection Manager (CM).

   For more information, see "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.

Configuring Time-Stamp Rounding for Purchase Grants

To configure BRM to round time stamps to midnight for the resources granted by purchase events:

1. Configure time-stamp rounding for cycle grants. See "Configuring Time-Stamp Rounding for Cycle Grants". You do not need to stop and restart the CM at the end of the procedure, although you may do so if you want.
2. Go to *BRM_home*/**sys/data/config**.
3. Use the following command to create an editable XML file from the rating instance of the **/config/business_params** object:

```
pin_bus_params -r BusParamsRating bus_params_rating.xml
```

   This command creates an XML file named **bus_params_rating.xml.out** in your current directory. If you do not want this file in your current directory, specify the path as part of the file name.

4. In **bus_params_rating.xml.out**, set **TimeStampRoundingForPurchaseGrant** to **enabled**:

Copyright © 2025, 2026, Oracle and/or its affiliates.

```
<TimeStampRoundingForPurchaseGrant>enabled</TimeStampRoundingForPurchaseGrant>
```

> **Caution**
>
> BRM uses the XML in this file to overwrite the existing instance of the **/config/business_params** object. If you delete or modify any other parameters in the file, these changes affect the associated aspects of the BRM configuration.

5. Save and exit the file.
6. Rename the **bus_params_rating.xml.out** file to **bus_params_rating.xml**.
7. Use the following command to load your changes into the **/config/business_params** object:

   ```
   pin_bus_params bus_params_rating.xml
   ```

   You should run this command from the *BRM_home***/sys/data/config** directory, which includes support files used by the utility. To run it from a different directory, see "pin_bus_params" in *BRM Developer's Guide*.
8. Read the object with the **testnap** utility or the Object Browser to verify that all fields are correct.

   For general instructions on using **testnap**, see "Using the testnap Utility to Test BRM" in *BRM Developer's Guide*. For information on how to use Object Browser, see "Reading Objects" in *BRM Developer's Guide*.
9. Stop and restart the CM.

   For more information, see "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.

About Noncurrency Sub-Balances That Start on First Usage

When a noncurrency resource is configured to start on first usage (when the customer first consumes the resource balance), BRM always creates a new sub-balance for that resource when it is granted. A new sub-balance is created for each resource even when a product or discount grants multiple first-usage resources of the same type. (For more information about first-usage start times, see "About Balance Impacts That Become Valid on First Usage").

For example, a product includes two grants of 100 free minutes. Each grant starts on first usage and ends at the end of the cycle. When the product is purchased, BRM adds two free-minute sub-balances to the account, one for each grant.

Keeping first-usage resources in separate sub-balances enables those balances to have unique validity periods, which are set individually when each resource is consumed. For example, if there are two balances of 100 free minutes each and the customer makes a 30-minute phone call, the validity period of only the balance that was consumed is set when the call is made.

A resource balance that has a first-usage start time will remain available for consumption for as long as the balance is not used. For example, on January 1, a customer purchases a deal that includes 30 free text messages that are valid for 30 days from first usage. The text message balance remains unused in the account until April 18, when the customer sends the first text message. The validity period of the text message balance is then set to start on April 18 and end 30 days later, on May 17.

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Configuring Sub-Balances

You can optionally configure sub-balances to track resources for more specific types of usage such as:

- Free minutes per call session.
- Frequent flyer miles per service instance.
- Friends and family calls to specific locations.
- Discount amounts shared with sponsored accounts.

You configure sub-balances for the entire BRM system by editing and loading a configuration file.

By default, common resources for all services owned by an account are placed in the same sub-balance if they have the same validity period and are associated with the same balance group. (You can create separate balance groups per service in Pricing Center. See "Tracking Resources by Service").

To limit the allocation and consumption of resources, you configure sub-balances by specifying the following values:

- *Resource ID*. This is the ID for the type of resource in the sub-balance, such as dollars or free minutes.
- Event type. This is the type of event that impacts the sub-balance, such as GSM usage events (**/event/session/telco/gsm**).
- *Contributors*. Contributors can be any field in the event record or object. There are two types of contributors:
  - *Retrieving contributor*. All sub-balances with common retrieving contributors are summed when sub-balances are retrieved. For example, to retrieve the total balance for specific services, specify the service object as the retrieving contributor.
  - *Updating contributor*. A sub-balance is created or updated by balance impacts for each unique updating contributor. For example, to add or deduct free minutes for specific dialup sessions, specify the session object as the updating contributor.

About Sub-Balance Contributors

Sub-balance contributors are specified by a field name from the event record or object. The value in the field you specify is used to retrieve and update the sub-balances. Some fields you might want to use as contributors include:

- The service object.

  Specify a service field to track resources for specific service instances such as fax, telephony, and text messaging.

  > **Note**
  >
  > Specifying the service object in the configuration is one way of creating service-level balances. The other way is to create a balance group for the service when you define your plans in Pricing Center. See "Tracking Resources by Service".

- The session object.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Specify a session field to track resources per session instance. This is useful when you offer discounts for certain levels of usage (for example, 10 frequent flyer miles per hour of phone calls).

- The product object.

  When several products in the same plan have different rollover rules, tracking resources per product permits BRM to update the free minutes for each product.

- The phone number field.

  Specify a phone number field to track resources for called telephone numbers.

- The account object.

  Specify the account field to track resources that are shared among several accounts, such as earned free minutes. When free minutes are distributed, they can be divided based on each account's usage level.

Retrieving and Updating Sub-Balances

Sub-balances are retrieved for a given resource. For example, when CSRs use Customer Center to view an account's current cash balance, all valid cash sub-balances (sub-balances with current validity periods and matching contributors) are summed to provide the current resource balance.

Sub-balances are updated by a balance impact. When a balance impact affects more than one sub-balance, the sub-balances are updated in the following order:

1. **By validity period**. By default, sub-balances are updated in chronological order based on the validity start date or end date. You specify the order in which sub-balances are used by setting up resource consumption rules. See "Specifying the Order in Which Resource Sub-Balances Are Consumed".
2. **By contributor**. Sub-balances with specific field contributors are impacted before sub-balances that accept all contributors (with an asterisk or empty contributor values). See "Configuring Sub-Balances".

Configuring Sub-Balances

To configure sub-balances, you edit the sub-balance configuration file and then run the **load_pin_sub_bal_contributor** utility to load the contents of the file into the **/config/sub_bal_contributor** object in the BRM database.

Editing the pin_sub_bal_contributor File

Each line in the **pin_sub_bal_contributor** file defines the usage for which a sub-balance is created. The configurations apply to all sub-balances in the BRM database, but they are implemented at the balance group level. That is, when a usage event occurs, if the account has more than one balance group, only the sub-balances within the specified balance groups are impacted.

To add sub-balance configurations, use this syntax:

```
resource_type:event_type:retrieving_contributor:updating_contributor
```

For example:

```
1000003:/event/session/gprs:PIN_FLD_SESSION_OBJ:PIN_FLD_SESSION_OBJ
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

where:

- **1000003** is the resource type for frequent flyer miles.
- **/event/session/gprs** is the event type that impacts the frequent flyer miles resource.
- **PIN_FLD_SESSION_OBJ** is the retrieving contributor. Therefore, a separate balance summary is retrieved for each unique GPRS session.
- **PIN_FLD_SESSION_OBJ** is the updating contributor. Therefore, separate sub-balances are created or updated for each unique GPRS session.

The contributors can be a specific field or an asterisk (*) to indicate any contributor. For example, the following entry retrieves and tracks dollars (resource type 840) in a single sub-balance for all events with any contributor:

```
840:/event:*:*
```

To configure several sub-balances for one resource type, leave subsequent resource fields empty. If no resource is specified, the previous resource specified is used. For example, the next two configurations use the same resource (100002):

```
100002 : /event/session/telco/gsm : PIN_FLD_SESSION_OBJ : PIN_FLD_SESSION_OBJ
       : /event/session/dialup : PIN_FLD_SERVICE_OBJ : PIN_FLD_SERVICE_OBJ
```

Running the load_pin_sub_bal_contributor Utility

> **Note**
>
> - When you run the **load_pin_sub_bal_contributor** utility, it replaces the existing sub-balance configurations. If you are updating a set of sub-balance configurations, you cannot load new configurations only. You load complete sets of sub-balance configurations each time you run the **load_pin_sub_bal_contributor** utility.
> - The resources specified in the **pin_sub_bal_contributor** file must exist in the BRM database before the **load_pin_sub_bal_contributor** utility is run. Therefore, you must first load the **pin_beid** file by running the **load_pin_beid** utility. See "load_pin_beid".

Use the following command to run the **load_pin_sub_bal_contributor** utility:

```
load_pin_sub_bal_contributor pin_sub_bal_contributor
```

If you are not in the same directory as the **pin_sub_bal_contributor** file, include the complete path to the file. For example:

```
load_pin_sub_bal_contributor BRM_home/sys/data/pricing/example/pin_sub_bal_contributor
```

For more information, see "load_pin_sub_bal_contributor".

To verify that the sub-balance configurations loaded correctly, display the **/config/sub_bal_contributor** object by using Object Browser, or use the **robj** command with the **testnap** utility.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Sub-Balance Configuration Example

The following examples show how sub-balances can be configured for various business needs.

Sub-Balances per Service Example

There are two ways to track resources per service instance:

- By creating a balance group for the service. See "Tracking Resources by Service".
- By specifying the service object as a contributor in the **pin_sub_bal_contributor** file.

Table 5-1 shows contributor configurations to track resources for GSM services. The resources are dollars and free minutes.

Table 5-1 Contributor Configurations

Resource	Event Type	Retrieving Contributor	Updating Contributor
Dollars	/event/session/gsm	PIN_FLD_SERVICE_OBJ	PIN_FLD_SERVICE_OBJ
Free minutes	/event/session/gsm	PIN_FLD_SERVICE_OBJ	PIN_FLD_SERVICE_OBJ
Free minutes	/event/cycle_forward	PIN_FLD_SERVICE_OBJ	PIN_FLD_SERVICE_OBJ
Free minutes	/event/cycle_forward	PIN_FLD_SERVICE_OBJ	PIN_FLD_SERVICE_OBJ

Each line in the above example specifies a configuration.

- **Dollars** is a currency sub-balance that is created in the account balance group.
- **/event/session/gsm** is the event type that will impact the dollars resource.
- **PIN_FLD_SERVICE_OBJ** is the retrieving contributor, so the dollars resource for each unique GSM service is summed when retrieving balance information.

  If you specify an asterisk instead of PIN_FLD_SERVICE_OBJ for the retrieving contributor, dollars for *all* GSM services owned by the account would be summed when retrieving balance information.
- **PIN_FLD_SERVICE_OBJ** is the updating contributor, so a separate sub-balance is created to track dollars for every unique GSM service that the account owns, such as telephony, fax, and SMS.
- **Free minutes** is a noncurrency sub-balance that is created in the account balance group.
- **/event/session/gsm** is the event type that will impact the free minutes resource.
- **PIN_FLD_SERVICE_OBJ** is the retrieving contributor, so the free minutes resource for each unique GSM service is summed when retrieving balance information.

  If you specify an asterisk instead of PIN_FLD_SERVICE_OBJ for the retrieving contributor, free minutes for *all* GSM services owned by the account would be summed when retrieving balance information.
- **PIN_FLD_SERVICE_OBJ** is the updating contributor, so a separate sub-balance is created to track free minutes for every unique GSM service that includes free minutes. Additional free minutes and consumption of free minutes are restricted to the GSM service instance.

Copyright © 2025, 2026, Oracle and/or its affiliates.

If you specify an asterisk instead of PIN_FLD_SERVICE_OBJ as the updating contributor, free minutes granted by every GSM service owned by the account would be shared by all the GSM services.

- **Free minutes** is a noncurrency sub-balance that is created in the account balance group.
- **/event/cycle_forward** is the event type that will impact the free minutes resource.
- **PIN_FLD_SERVICE_OBJ** is the retrieving contributor, so free minutes granted by cycle forward events are summed for each unique service when retrieving balance information.
- **PIN_FLD_SERVICE_OBJ** is the updating contributor, so a separate sub-balance is created to track free minutes granted by cycle forward events for each unique service that grants free minutes. Consumption of free minutes granted at the beginning of the cycle is restricted to the service that granted them.

Specifying the Order in Which Resource Sub-Balances Are Consumed

Customers are sometimes granted multiple sub-balances of a particular resource, such as minutes. For example, a customer's balance might include minutes granted at the start of the accounting cycle and rollover minutes from the previous month. Because the minutes have different validity periods, they are grouped into different resource sub-balances. See "About Noncurrency Sub-Balances".

When the customer uses a service, BRM needs to know which minutes (or sub-balance) to use first. You use resource consumption rules to specify the order in which resource sub-balances are consumed, according to the validity start time and end time.

For example, to use rollover minutes first, you configure BRM to consume sub-balances based on the earliest validity start time. To use the minutes that expire first, you configure BRM to consume sub-balances based on the earliest validity end time.

Consumption Rule Descriptions

BRM supports the resource consumption rules shown in Table 5-2:

Table 5-2 Supported Consumption Rules

Consumption Rule	Description
Earliest start time (EST)	Consume the sub-balance with the earliest validity start time first.
Latest start time (LST)	Consume the sub-balance with the latest validity start time first.
Earliest expiration time (EET)	Consume the sub-balance with the earliest validity end time first.
Latest expiration time (LET)	Consume the sub-balance with the latest validity end time first.
Earliest start time and latest expiration time (ESTLET)	Consume the sub-balance with the earliest validity start time first. If multiple sub-balances have the same start time, use the one with the latest end time first.
Earliest start time and earliest expiration time (ESTEET)	Consume the sub-balance with the earliest validity start time first. If multiple sub-balances have the same start time, use the one with the earliest validity end time first.
Latest start time and earliest expiration time (LSTEET)	Consume the sub-balance with the latest validity start time first. If multiple sub-balances have the same validity start time, use the one with the earliest validity end time first.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 5-2 (Cont.) Supported Consumption Rules

Consumption Rule	Description
Latest start time and latest expiration time (LSTLET)	Consume the sub-balance with the latest validity start time first. If multiple sub-balances have the same validity start time, use the one with the latest validity end time first.
Earliest expiration time and earliest start time (EETEST)	Consume the sub-balance with the earliest validity end time first. If multiple sub-balances have the same validity end time, use the one with the earliest validity start time first.
Earliest expiration time and latest start time (EETLST)	Consume the sub-balance with the earliest validity end time first. If multiple sub-balances have the same validity end time, use the one with the latest validity start time first.
Latest expiration time and earliest start time (LETEST)	Consume the sub-balance with the latest validity end time first. If multiple sub-balances have the same validity end time, use the one with the earliest validity start time first.
Latest expiration time and latest start time (LETLST)	Consume the sub-balance with the latest validity end time first. If multiple sub-balances have the same validity end time, use the one with the latest validity start time first.

For example, if a customer's balance includes the following minutes and the customer makes a phone call on February 10, the consumption rules specify which group of minutes are impacted first:

- 100 Anytime Minutes with a validity period of February 1 to February 28.
- 50 rollover Anytime Minutes with a validity period of January 1 to February 28.
- 200 bonus Anytime Minutes with a validity period of January 15 to June 15.

If the consumption rule is set to earliest start time (EST), BRM applies the balance impact to the 50 rollover Anytime Minutes first. Likewise, if the rule is set to earliest expiration time and latest start time (EETLST), BRM applies the balance impact to the 100 Anytime Minutes first.

You specify the order in which resource sub-balances are consumed in your price plans. You can also set systemwide and default resource settings. BRM reads and uses the consumption rule settings in the order shown below:

1. **Loan consumption rules**. If a loan sub-balance is present, it will be consumed first. If there are multiple loan sub-balances, they are consumed according to the existing consumption rules.
2. **Price plan setting**. Your price plans can include resource-to-consumption rule mappings for each service that you support. This enables you to have different consumption rules for the same resource based on the plans owned by the customer. When a customer purchases a plan, the plan's resource-to-consumption rule mapping is stored in the customer's **/balance_group** object. If there are any conflicting rules for the same resource, BRM uses the rule from the most recently purchased plan.
3. **Systemwide resource setting**. You can specify a systemwide resource-to-consumption rule mapping for each service that you support. BRM uses the systemwide settings only if a rule is not defined for the resource in the customer's purchased plans. BRM stores systemwide settings in the **/config/beid** object.
4. **Default resource setting**. You can specify a default setting that applies to all resources. This setting is used only if a consumption rule is not defined for the resource in the customer's purchased plans or if there is not a systemwide resource setting. BRM stores the default resource setting in the **/config/business_params** object.

Copyright © 2025, 2026, Oracle and/or its affiliates.

How BRM Applies Consumption Rules

To apply resource consumption rules, BRM automatically orders an account's resource sub-balances whenever the account adds or changes a resource. For example, if you modify a resource with a consumption rule of EET, BRM orders the resource sub-balances based on the validity end time, starting with the earliest expiration time first. When the account has a balance impact, BRM searches through the account sub-balances, in order, and applies the balance impact to the first sub-balance that matches the following criteria:

- Has a validity period that matches the event's time stamp.
- Has a balance to consume.

If there is any remaining balance impact, BRM applies it to the next sub-balance that matches the criteria. This process continues until all of the sub-balances have been depleted. BRM then restarts the search from the beginning and applies any remaining balance impact to the first sub-balance that matches the validity period.

For example, assume that an account with the following resource sub-balances makes a 30-minute phone call on June 4. If the consumption rule is set to LSTEET, BRM first consumes the 5 minutes from sub-balance A and then consumes the 10 minutes from sub-balance C. Because the account's sub-balances have been depleted, BRM charges the remaining 15 minutes to sub-balance A.

- Sub-balance A has a balance of 5 minutes with a validity period of June 1 to June 15.
- Sub-balance B has a balance of 0 minutes with a validity period of June 1 to June 30.
- Sub-balance C has a balance of 10 minutes with a validity period of May 1 to July 15.
- Sub-balance D has a balance of 0 minutes with a validity period of January 1 to December 30.

The method BRM uses to implement resource consumption rules is different for batch rating and real-time rating.

How Batch Rating Applies Consumption Rules

In batch rating, resource consumption rules are applied by the DAT_BalanceBatch module. The DAT_BalanceBatch module orders resource sub-balances when Pipeline Manager starts and when an account adds or modifies a resource. When processing CDRs, FCT_ApplyBalance uses the DAT_BalanceBatch module to search through an account's resource sub-balances, in order, and find the first sub-balance that matches the CDR time stamp and has a balance to consume.

By default, Pipeline Manager supports all of your price plan, systemwide, and default consumption rule settings. However, you can configure Pipeline Manager to use the default setting only by using the **UseFlexibleConsumptionRule** registry entry in the DAT_BalanceBatch module:

- **True**: Uses the consumption rules defined for each resource in a balance group. If a consumption rule is not defined, it uses the rules defined in the **/config/beid** object. If a consumption rule is not defined in a balance group or in the **/config/beid** object, the module uses the rule defined in the **multi_bal** instance of the **/config/business_params** object.
- **False**: Uses the systemwide consumption rule defined in the **multi_bal** instance of the **/config/business_params** object only.

Copyright © 2025, 2026, Oracle and/or its affiliates.

How Real-Time Rating Applies Consumption Rules

In real-time rating, consumption rules are applied by the Balance FM opcodes. When an account is created or modified, the Balance FM opcodes read the consumption rule for each resource and order the resource sub-balances appropriately. When a customer has a balance impact, the Balance FM opcodes search through the customer's resource sub-balances, in order, and apply the balance impact to the first sub-balance that matches the event time stamp and has a balance to consume.

Setting Resource Consumption Rules

To assign resource consumption rules, perform one or more of these tasks:

- Setting Consumption Rules in Your Price Plans
- Setting Systemwide Consumption Rules for Each Resource
- Setting the Default Consumption Rule

For information about changing your consumption rule settings, see "Modifying Resource Consumption Rules".

Setting Consumption Rules in Your Price Plans

You set resource consumption rules in your price plans by using Pricing Center. See the Pricing Center Help for more information.

> **Note**
>
> You can also use the "loadpricelist" utility to assign resource consumption rules in your price plans. You specify the rules in the **price_list.xml** file's **consumption_rule** field and then load the file into the database by using the **loadpricelist** utility. See "Using the XML Pricing Interface to Create a Price List ".

Pricing Center and **loadpricelist** store your price lists and consumption rule settings in **/plan** objects in the BRM database. When a customer purchases a product, the plan's resource-to-consumption rule mapping is stored in the customer's **/balance_group** object.

If you use a custom client application to create price lists, you must modify your application to pass consumption rule settings to the Pricing FM opcodes. For more information, see "Customizing Credit Limit and Consumption Rules" in *BRM Opcode Guide*.

Setting Systemwide Consumption Rules for Each Resource

You define systemwide consumption rules for each resource that you support by using Resource Editor. See the Resource Editor Help for more information.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> You can also use the **pin_beid** configuration file to assign systemwide resource consumption rules. You specify the rules in the file's **con_rule_id** column and then load the file into the database by using the **load_pin_beid** utility. See "load_pin_beid" for more information.

Resource Editor and **load_pin_beid** store the systemwide consumption rule setting for each resource in the **/config/beid** object.

Setting the Default Consumption Rule

The default consumption rule applies to all resources in your system. The default setting is earliest start time and earliest expiration time (ESTEET). You can change this setting by modifying a field in the **multi_bal** instance of the **/config/business_params** object created during BRM installation.

You modify the **/config/business_params** object by using the **pin_bus_params** utility.

To set the default resource consumption rule:

1. Use the following command to create an editable XML file for the **multi_bal** class:

   ```
   pin_bus_params -r BusParamsMultiBal bus_params_multi_bal.xml
   ```

   This command creates the XML file named **bus_params_multi_bal.xml.out** in your working directory. If you do not want this file in your working directory, specify the path as part of the file name.

2. Search the XML file for following line:

   ```
   <SortValidityBy>ESTEET</SortValidityBy>
   ```

3. Change **ESTEET** to the appropriate value:

   - Earliest start time (EST)
   - Latest start time (LST)
   - Earliest expiration time (EET)
   - Latest expiration time (LET)
   - Earliest start time and latest expiration time (ESTLET)
   - Earliest start time and earliest expiration time (ESTEET)
   - Latest start time and earliest expiration time (LSTEET)
   - Latest start time and latest expiration time (LSTLET)
   - Earliest expiration time and earliest start time (EETEST)
   - Earliest expiration time and latest start time (EETLST)
   - Latest expiration time and earliest start time (LETEST)
   - Latest expiration time and latest start time (LETLST)

   For a description of each setting, see "Consumption Rule Descriptions".

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> BRM uses the XML in this file to overwrite the existing **multi_bal** instance of the **/config/business_params** object. If you delete or modify any other parameters in the file, these changes affect the associated aspects of the BRM billing configuration.

4. Save and exit the file.
5. Rename the **bus_params_multi_bal.xml.out** file to **bus_params_multi_bal.xml**.
6. Use the following command to load the change into the **/config/business_params** object:

   ```
   pin_bus_params bus_params_multi_bal.xml
   ```

   You should run this command from the *BRM_home***/sys/data/config** directory, which includes support files used by the utility.
7. Read the object with the **testnap** utility or Object Browser to verify that all fields are correct.

You do not need to restart the Connection Manager (CM) to update this entry.

Modifying Resource Consumption Rules

You can modify the resource consumption rules that you set in your price plans, in your **/config/beid** object, and in your **/config/business_params** object at any time. However, the new and changed settings apply to new customers only. Your existing customers will continue to use the previous settings:

- Changes to plan-level consumption rules do not trigger any update to existing customers that own the changed plan. Existing plan owners continue to use the plan's previous settings.
- Changes to the **/config/beid** object do not trigger changes to existing customers. Existing customers continue to use the previous systemwide settings.
- Changes to the **/config/business_params** object do not trigger changes to existing customers. Existing customers continue to use the previous default setting.

> **Note**
>
> Pipeline Manager cannot use the new or modified **/config/beid** or **/config/business_params** consumption rule settings until they are loaded into pipeline memory. See "Reloading Data into a Pipeline Manager Module" in the *BRM Pipeline Manager Administration Guide*.

About Rollovers

Use the rollover feature to specify the quantity of unused resources that can be rolled over into subsequent cycles.

For example, you can configure a product so that a portion of unused free minutes from each cycle can be rolled over for use in the following cycle or cycles.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> Rollover resources are distinct from resources issued through one-time grants, such as those issued by a CSR or a discount product. One-time grants are valid between two specific dates.

You set up rollovers in a product's event map in Pricing Center.

By default, rollover events are included in the **pin_event_map** file.

About Rollover Resource Sub-Balances

BRM maintains each rollover resource as a sub-balance. Each sub-balance specifies a resource balance that is valid (available for use) between valid-from and valid-to dates stored in BRM. The total amount of a resource available to a customer for each cycle is equal to the sum of all resource sub-balances that are valid during the cycle.

BRM validates whether a sub-balance can be rolled over by checking the rules that govern rollover (such as the amount to roll over, the maximum rollover amount allowed, the maximum number of cycles to roll over, and so on).

When a rollover event occurs, one of three things happens to the balance in a currently valid resource sub-balance:

- **If the full amount is eligible for rollover**, BRM does one of the following:
  - Creates a new rollover sub-balance for the rolled over amount. The rollover sub-balance validity period has the same valid-from date as the original sub-balance, and its valid-to date is extended to the end of the new cycle.
  - Adds the rolled over amount to an existing sub-balance if the existing sub-balance has the same data as would the new rollover sub-balance (such as the resource type, rollover rules, valid-from and valid-to dates, sub-balance contributors, and so on).
- **If only a portion of the resources is eligible for rollover**, the amount in the sub-balance is divided into a non-rollover sub-balance and a rollover sub-balance. The non-rollover sub-balance has the same valid-to date as the original sub-balance. Its resource balance is used for late-arriving usage events. The valid-to date for the rollover sub-balance is extended to the end of the new cycle. Its resources are available for the customer to use in the new cycle.
- **If none of the resource is eligible for rollover**, the sub-balance is not changed. This condition occurs when the sub-balance has already been rolled over the maximum number of times allowed. Amounts in this sub-balance are available only for late-arriving usage events.

> **Note**
>
> Your system's memory limits might limit the number of months you can roll over a sub-balance. The number of resource sub-balances that BRM must maintain varies according to the number of rollover cycles allowed. You set the maximum allowed rollover cycles when configuring the rollover balance impact in Pricing Center.

CSRs can view rollover resource balances by using Customer Center.

Copyright © 2025, 2026, Oracle and/or its affiliates.

When Rollover Events Occur

Resource sub-balances can be rolled over to another cycle as soon as they surpass the valid-to date (expire). Most resource sub-balances expire on the account's billing day of month (DOM). However, some sub-balances may expire in the middle of a cycle due to a product or service cancellation or due to flexible cycles.

Because many BRM features depend on sub-balances being rolled over the day after they expire, BRM rolls over resources at the following times:

- **At the end of the billing cycle**: BRM automatically rolls over all eligible resource sub-balances to the next cycle as part of the billing process. This catches all resource sub-balances that expire on the account's billing DOM.
- **When you run the pin_bill_day script**: The **pin_bill_day** script automatically runs the **pin_rollover** utility as part of the billing process. The **pin_rollover** utility rolls over any resource sub-balances that have expired but have not been rolled over to the next cycle. This catches any resource sub-balances that expire in the middle of the cycle.

Deleting Expired Sub-Balances

To delete expired sub-balances, use the **pin_sub_balance_cleanup** utility.

Rollover Example

The following example demonstrates:

- Granting resources (free minutes) at the start of each cycle (calendar month).
- Rolling over unused free minutes from previous months at the start of each month.

> **Note**
>
> For information about rolling over resources that expire in the middle of a cycle, see "About Rolling Over Resources That Expire in Midcycle".

- Dividing an initial resource grant into rollover and non-rollover sub-balances.
- Limiting the total number of free minutes that can be rolled over from previous months.
- Dividing a rollover balance into rollover and non-rollover sub-balances when the Maximum Cumulative Rollover Total value is reached.
- The default order in which free minutes are consumed from the valid resource sub-balances.
- Expiration of free minutes in a sub-balance when the sub-balance has been rolled over the number of cycles specified by the Maximum Number of Rollover Cycles value.

In this example, the customer's product includes:

- 500 free minutes granted at the beginning of each usage cycle.
- A rollover specifying that:
  - Up to 100 unused free minutes from each month can be rolled over.
  - The maximum number of rollover cycles is 2.

Copyright © 2025, 2026, Oracle and/or its affiliates.

– A limit of 150 *total* rollover minutes from previous months can be rolled over into a new month.

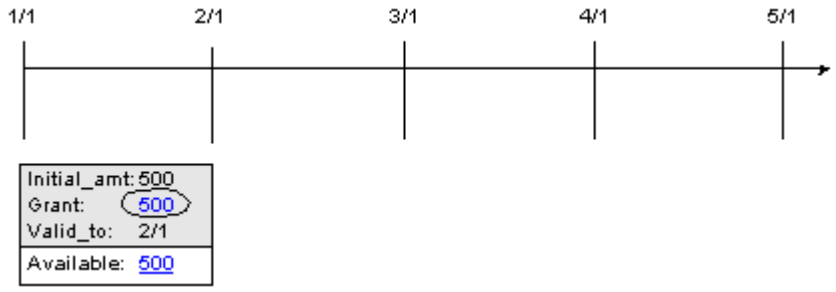
Other assumptions:

- Cycles are monthly and start on the first day of each calendar month.
- The product with the rollover is valid starting January 1.
- The customer consumes no free minutes until March.

> **Note**
>
> In this example:
>
> – New sub-balances, or those that have a change in their validity period, are highlighted in gray.
>
> – New grant resource values or changes to current resource amounts within the sub-balances are in **bold black**.
>
> – Components of available resource totals are in **blue**.
>
> – Components of used resource totals are in **red**.

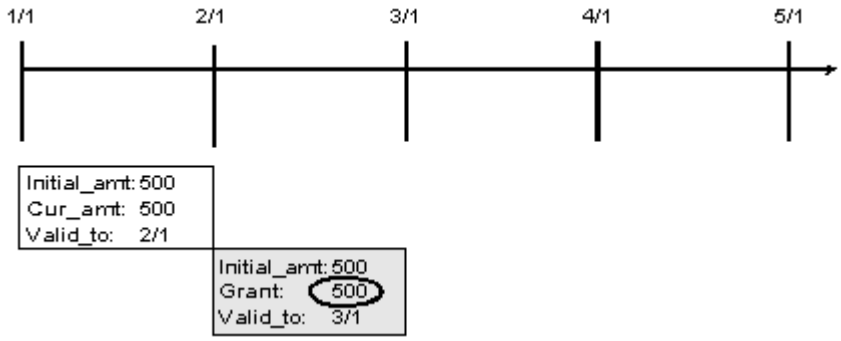
1. On January 1, the customer is granted 500 free minutes for use during January as shown in Figure 5-1. The free minutes are maintained in a resource sub-balance.

Figure 5-1 Initial 500 Free Minutes Grant in January



2. On February 1:
   - The cycle forward event creates a new resource sub-balance to track the grant of 500 free minutes for use during February as shown in Figure 5-2.

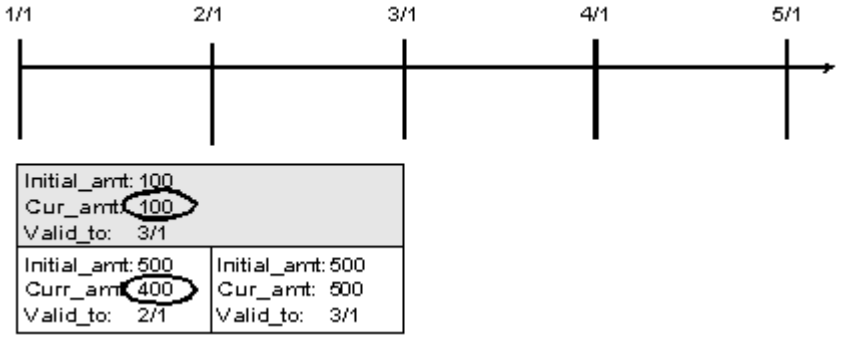
Figure 5-2 February Free Minutes Grant



Copyright © 2025, 2026, Oracle and/or its affiliates.

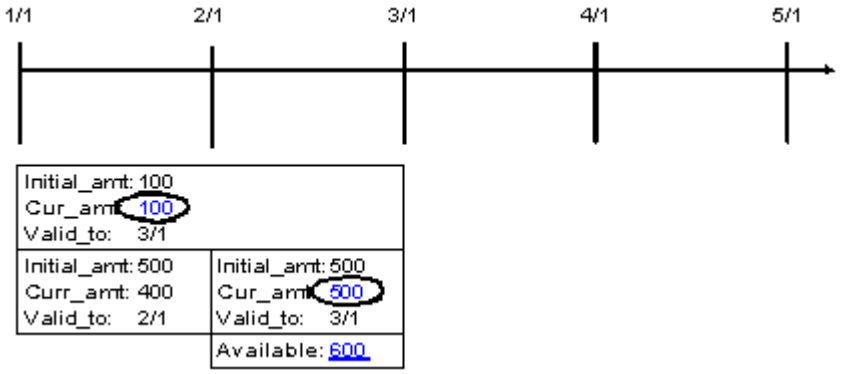
- The cycle rollover event creates a new resource sub-balance for tracking unused January free minutes that roll over into February. 100 minutes are put in the new sub-balance, and they are valid until March 1. The original sub-balance for January is decremented 100 minutes, leaving 400 minutes available for late-arriving January events as shown in Figure 5-3.

Figure 5-3 January Free Minutes Rollover



The customer now has 600 free minutes (500 granted February 1 plus 100 rolled over from January) available for use during February as shown in Figure 5-4.

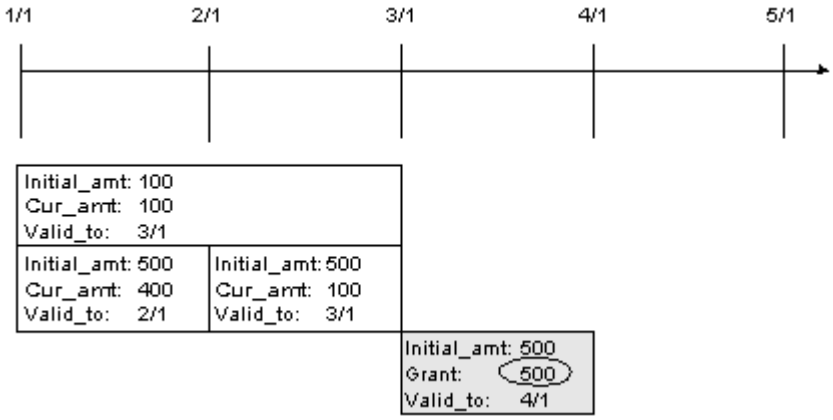
Figure 5-4 February Total Free Minutes Available



3. On March 1:
   - The cycle forward event creates a new resource sub-balance to track the grant of 500 free minutes for use during March as shown in Figure 5-5.

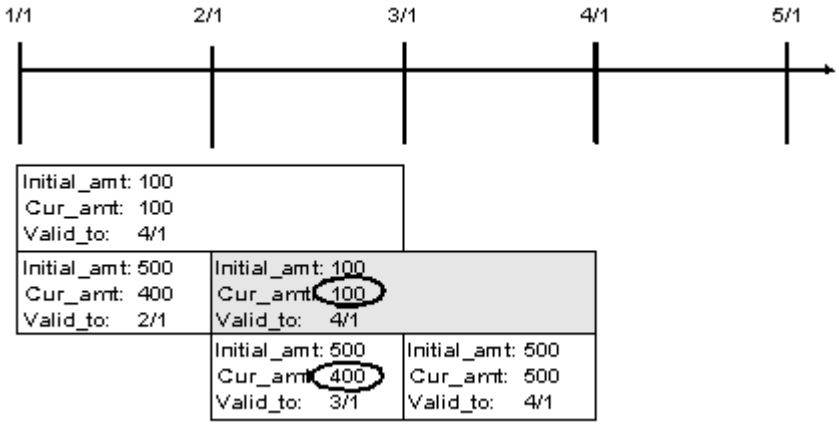
Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 5-5 March Free Minutes Grant



- The cycle rollover event creates a new resource sub-balance for tracking unused February free minutes that roll over into March as shown in Figure 5-6. 100 minutes are put in the new sub-balance, and they are valid until April 1. The original sub-balance for February is decremented by 100 minutes, leaving 400 minutes available for late-arriving February events.

Figure 5-6 February Free Minutes Rollover

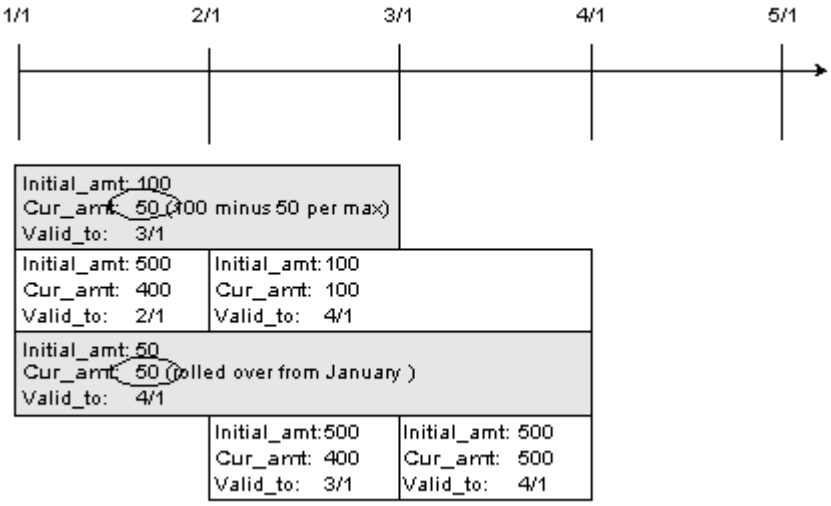


- The cycle rollover event divides the January rollover minutes into two sub-balances to enforce the rule that only 150 *total* rollover minutes for a resource can carry forward into a new month. Because 100 free February minutes are already rolled over, only 50 minutes can be rolled over from the January rollover sub-balance for use in March.

  One new sub-balance contains 50 minutes available for use for late-arriving February and January events. The other contains 50 rollover minutes that the customer can use in March. The resource balances for each month are shown in Figure 5-7.

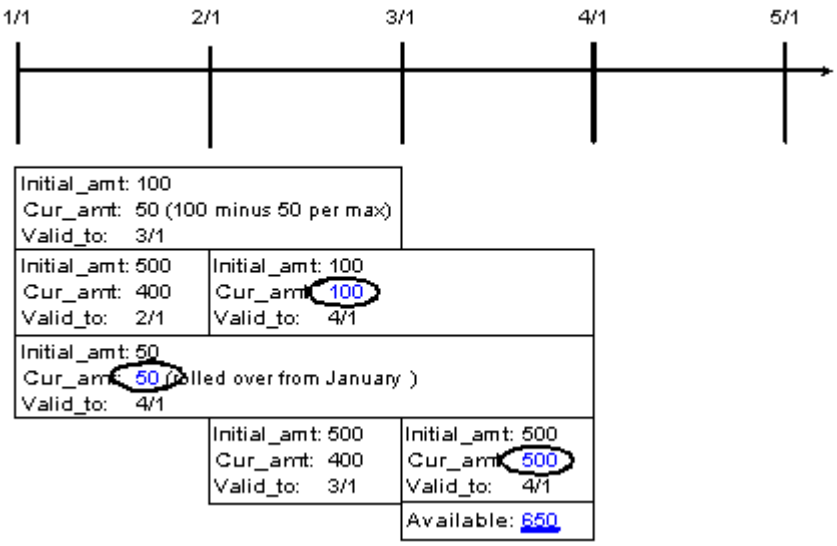
Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 5-7 Remaining January Free Minutes Rollover



The customer now has 650 free minutes (500 granted March 1 plus 100 rolled over from February plus 50 rolled over from January) available for use during March as shown in Figure 5-8.

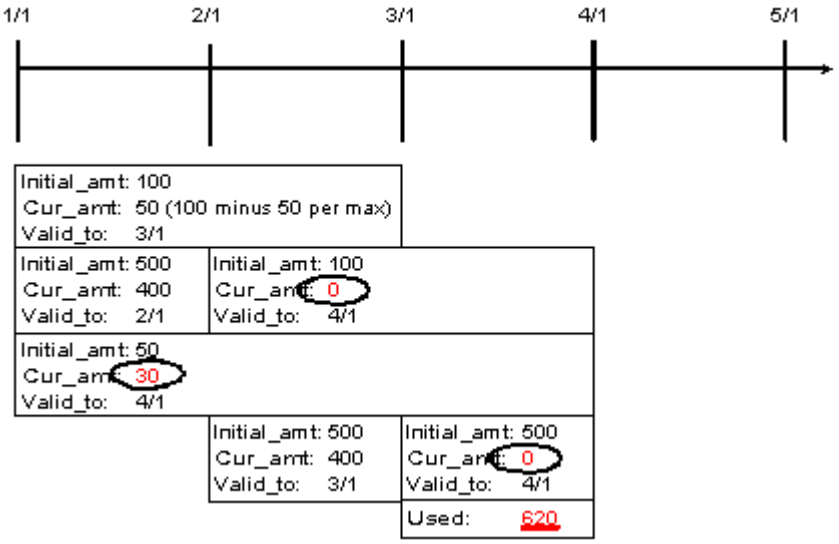
Figure 5-8 March Total Free Minutes Available



4. During March, the customer consumes 620 minutes as shown in Figure 5-9. Sub-balance resources are used starting with the newest sub-balance, as indicated by the sub-balance valid-from dates:
   - All 500 free minutes from the March grant are consumed, leaving a zero balance.
   - All 100 February rollover minutes are consumed, leaving a zero balance.
   - 20 of the 50 January rollover minutes are consumed, leaving 30.

Copyright © 2025, 2026, Oracle and/or its affiliates.

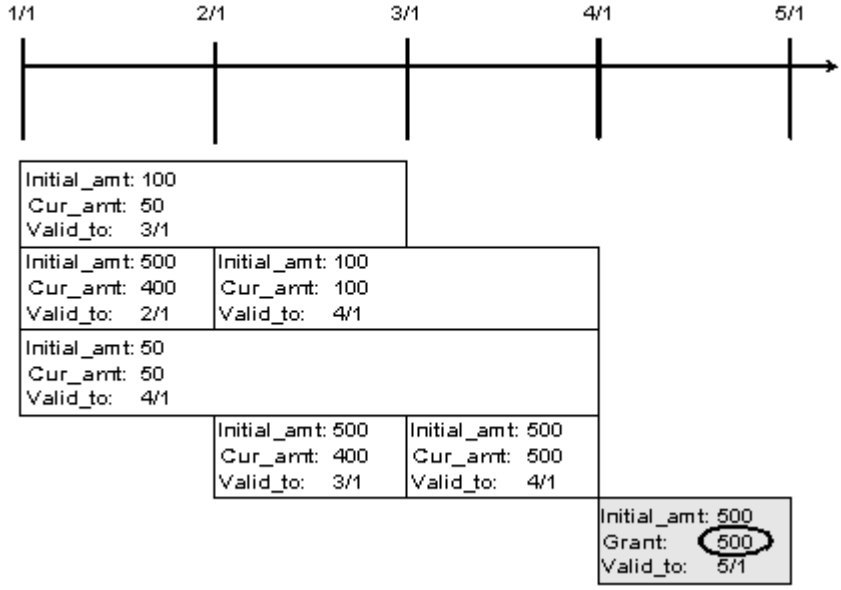
Figure 5-9 March Free Minutes Usage



5. On April 1, the cycle forward event creates a new resource sub-balance to track the grant of 500 free minutes for use during April as shown in Figure 5-10.

   No free minutes roll over into April from the March and February rollovers because these resources were consumed in March. The 30 free minutes remaining from the January rollover are not rolled over into April because the maximum number of cycles a grant can be rolled over is set to 2.

Figure 5-10 April Free Minutes Grant



The customer has only 500 free minutes available for April.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> - Rollover sub-balances are not truncated or prorated when the corresponding product is canceled.
> - If two products contributing to the same balance group have rollover rateplans configured for the same resource with the same rollover frequency, either of the products can be used to roll over the resources. In this case, rollover results may vary depending on the product that is selected first.

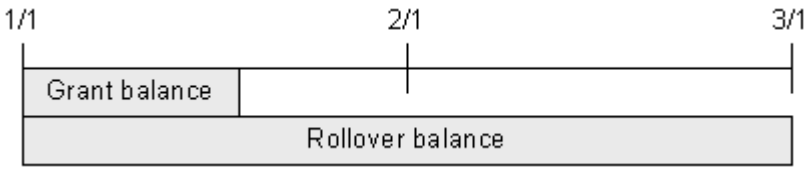
About Rolling Over Resources That Expire in Midcycle

A resource balance can expire in the middle of a cycle: for example, when the resource is valid for only minutes, hours, or days or when the resource is valid for one or more months and starts in the middle of a month.

If a resource's validity period does not end when the cycle ends, the rollover sub-balance is valid for the entire cycle in which the resource is granted and for the whole of the next cycle.

In Figure 5-11, a monthly cycle event grants 60 free minutes that are valid for two weeks from the grant date. The free minutes are granted on January 1. On January 14, when the balance of free minutes expires, any remaining resources that can be rolled over are added to a new sub-balance, which is valid from January 1 to March 1:

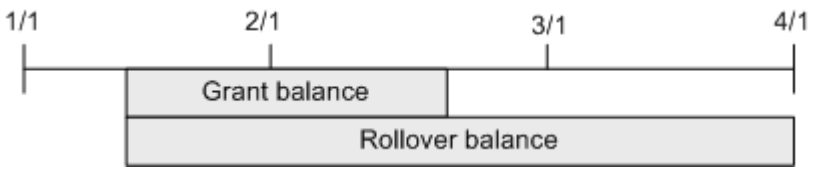
Figure 5-11 Grant and Rollover Balance Date Validity Example 1



The same rollover period applies when the validity period of the free minutes ends after the cycle in which they are granted. In Figure 5-12, a monthly cycle event grants 300 free minutes that are valid for six weeks from the grant date. The free minutes are granted on January 1. On February 11, when the balance expires, any remaining resources that can be rolled over are added to a new sub-balance, which is valid from January 1 to March 1:

If the resource's validity period does not start at the beginning of the last cycle, the resources are rolled over for one entire cycle.

Figure 5-12 Grant and Rollover Balance Date Validity Example 2



Copyright © 2025, 2026, Oracle and/or its affiliates.

Prorating Rollover Resources

Customers typically purchase and cancel products at some point in the middle of a cycle. When you set up a rollover in Pricing Center, you can specify whether rollover resources are prorated for the first and last cycles based on the number of days in the cycles that the product is owned.

Table 5-3 describes the rollover proration options:

Table 5-3 Rollover Proration Options

Proration Option	Description
Rollover entire amount	Roll over the available monthly rollover resources.
No rollover	Do not roll over any available monthly rollover resources.
Prorate rollover amount	Calculate the rollover resources based on the percentage of the cycle that the customer owned the rollover.

If you choose to prorate the rollover amount, BRM uses the equation in Figure 5-13 to calculate the amount to rollover:

Figure 5-13 Calculation for Amount to Rollover

$$\frac{ValidFrom - ValidTo}{DaysInCycle} \times Rollover$$

where:

- *ValidFrom* is the validity period's starting date. For example, if the validity period is from March 15 to April 14, ValidFrom is March 15.
- *ValidTo* is the validity period's ending date. For example, if the validity period is from March 15 to April 14, ValidTo is April 14.
- *DaysInCycle* is the number of days in the current cycle. For example, if the validity period is from March 15 to April 14, DaysInCycle is 31.
- *Rollover* is the available monthly rollover resource, such as 100 minutes.

For example, suppose:

- A customer buys a product on January 15 that grants 500 free minutes of use during each cycle.
- A rollover is set up so that up to 200 unused free minutes roll over into the following cycle.
- The usage cycle is a calendar month.
- The customer does not use any minutes during the first cycle.

Table 5-4 describes how the rollover is handled, depending on the Purchase Mid-cycle Proration setting:

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 5-4 Impact of Proration Options

Proration Option	Amount Rolled Over into February
Rollover entire amount	200 free minutes are rolled over from January into February. In February, the customer has 700 free minutes (200 rolled over plus the 500 free minutes February grant).
No rollover	No free minutes are rolled over from January into February. In February, the customer has only the February grant of 500 free minutes.
Prorate rollover amount	Approximately 110 free minutes are rolled over from January into February: (17/31) * 200 = 109.67 In February, the customer has about 610 free minutes (110 rolled over plus the 500 free minutes February grant).

A similar set of results applies to the last-month proration calculation specified in the Cancel Mid-cycle Proration setting.

About Rolling Over Free Resources during Plan Transition

When your customers transition from one plan to another, you can specify that free noncurrency resources be rolled over from one plan to another if both plans belong together to at least one plan list.

For more information on plan transitions, see "Transitioning between Plans".

To specify free resources rollover between plans during plan transition, you must perform these tasks:

- If you use a custom application for customer management, provide the trigger for controlled rollover in the input flist to the PCM_OP_SUBSCRIPTION_TRANSITION_PLAN opcode.
- When you create your plan lists, ensure that plans between which you want to allow rollovers are in at least one plan list together.

  For more information, see "About Plan Lists".

These general rules apply to controlled rollovers:

- Controlled rollovers are not affected by the product cycle rollover settings.

  For more information on cycle rollovers, see "About Rollovers".
- A controlled rollover does not count as a rollover and hence is not restricted to the maximum rollover quantity and the rollover units per period settings for the cycle rollover.
- The cycle grants of free resources that have been prorated during the plan transition are rolled over to the next plan.

Copyright © 2025, 2026, Oracle and/or its affiliates.

6

About Real-Time Rating

This chapter describes how to perform Oracle Communications Billing and Revenue Management (BRM) real-time rating if you are using Pricing Center for creating price lists.

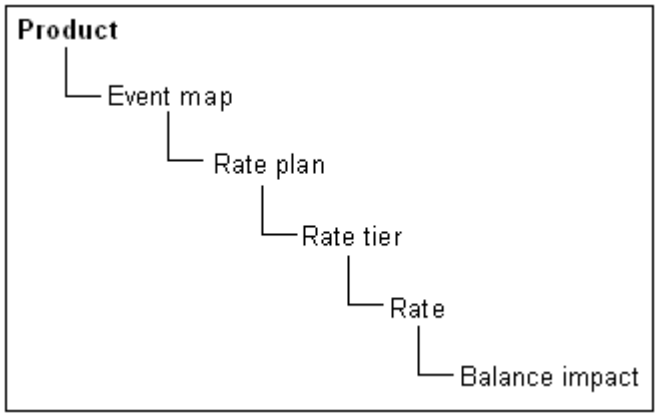
Topics in this document:

- About Organizing Real-Time Rates in a Product
- About Real-Time Rates and Balance Impacts
- About Fold Events
- Ensuring That All of an Event Is Rated

About Organizing Real-Time Rates in a Product

You organize rates in a product in a hierarchical structure as shown in Figure 6-1:

Figure 6-1 Product Rate Structure



This structure enables you to create multiple components at each level; for example, you can rate multiple events in a single product, create multiple rates based on dates, and create multiple balance impacts for a rate based on previously rated quantities.

About Real-Time Rate Plans

Usually, you use one rate plan per event. A rate plan includes the following:

- The rate or rates that apply to the event
- The currency to use for the event

  By default, rate plans use the system currency.
- Tax information, such as when to calculate the tax, and the tax code

  Assigning tax codes to rate plans enables you calculate taxes more accurately.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- For cycle forward fees, billing in advance

  See "Charging Cycle Forward Fees in Advance".

You can use multiple rate plans to choose a rate plan based on event data (for example, telephone call origins and destinations). See "Real-Time Rating Using Event Attributes".

In rare cases, you might need to use custom event analysis code to specify how to rate an event. See "About Custom Event Analysis".

You use rate plans to define how much to charge for an event. Rate plans include a currency, a tax code, and at least one rate tier. See "About Organizing Real-Time Rates in a Product".

> **Note**
>
> - Rate plan names and descriptions (for example, the names of rate plans, rate tiers, and date ranges) can include a maximum of 255 characters.
> - If you use pipeline batch rating to rate events, you must use the same rate plan names in your real-time rate plans and in the pipeline rate plans.

Specifying the Rate Plan Currency

You specify the currency in which each customer's fees are charged. See "Managing system and account currencies" in *BRM Managing Customers*.

Specifying the Rate Plan Tax Code

You can assign a tax code to enable BRM or a third-party tax calculation package to calculate taxes. See "Creating Tax Codes" in *BRM Calculating Taxes*.

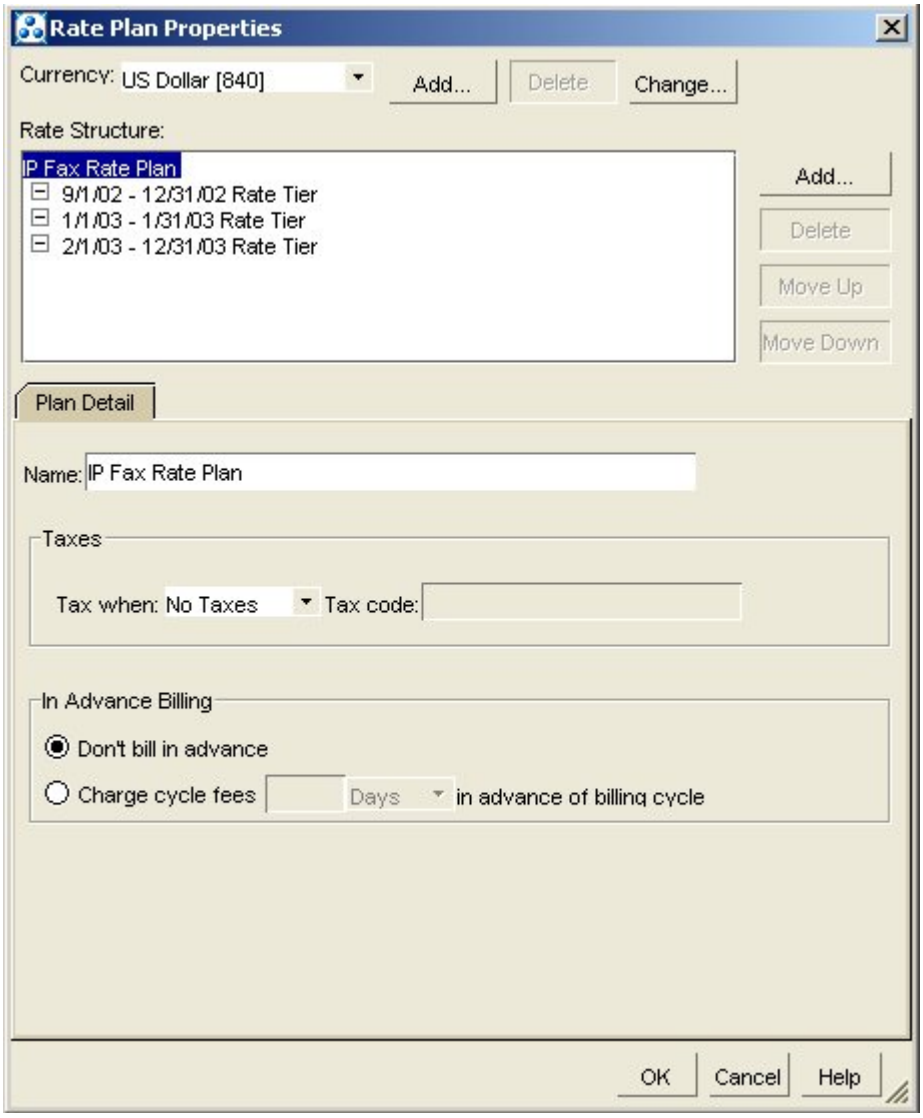
About Rate Tiers

You use rate tiers to define when and for how long rates are valid and which rates are applied first.

Figure 6-2 shows a rate plan that includes three rate tiers. Each of the rate tiers is valid for a range of dates:

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 6-2 Rate Plan with Three Rate Tiers



For each rate tier, you can specify one of the following:

- **Absolute date range**. This rate tier is in effect only in the dates specified (for example, March through September).
- **Relative date range**. This rate tier is in effect only in a range of days relative to the product purchase (for example, for 30 days after the purchase date).

> **Note**
>
> For either absolute or relative date ranges, you can specify that a rate tier's valid period starts immediately and never ends. In effect, this makes the rate tier always valid.

You need more than one rate tier only in the following cases:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- To apply a balance impact for different resources in a specific order. For example, if you give customers free hours, you can prioritize rate tiers to charge for hours before currency.
- If you create rates based on date or time. For more information about rate tiers, see "Real-Time Rating Based on Date and Time".

About Real-Time Rates and Balance Impacts

A rate defines the balance impact that results when the event is rated (for example, $2 for a one-hour dialup session). For information about balance impacts, see "Specifying How an Event Affects an Account Balance".

You usually use only one balance impact per rate, but you can specify multiple balance impacts for a single rate. For example:

- $1 per fax
- 10 points toward free faxes

In addition, you can group balance impacts in terms of quantity. For example:

- For the first 10 faxes:
  - $1 per fax
  - 10 points toward free faxes
- For the next 90 faxes:
  - 50 cents per fax
  - 25 points toward free faxes
- For 100 or more faxes:
  - 5 cents per fax
  - 50 points toward free faxes

For more information, see "Real-Time Rating Based on Event or Resource Quantity".

Specifying How an Event Affects an Account Balance

You specify the impact each event has on an account balance. You define each balance impact in terms of resources, general ledger IDs, proration, sponsorship, and discounts. In addition, you can assign an impact category to each balance impact.

The balance impact is the effect that rating has on an account balance. A balance impact can be positive or negative:

- A positive, or *credit*, impact is a charge to the customer (for example, $2 for an hour of Internet usage). The customer owes you the amount of the credit balance impact.
- A negative, or *debit*, impact is a resource given to the customer (for example, 10 free Internet access hours per month). You owe the customer the amount of the debit balance impact.

About Scaled Impacts

An event that uses a *scaled* resource amount impacts an account balance immediately when the event starts and accumulates until the event ends. For example, BRM starts rating an IP telephony call as soon as it starts. This enables BRM to track the impact to the account as the event is rated.

Copyright © 2025, 2026, Oracle and/or its affiliates.

A scaled amount affects a resource balance by multiplying the scaled amount value by the event quantity. For example:

Scaled rate of $1 per hour **X** event quantity of 10 hours **=** balance impact of $10

Use scaled impact when a resource is calculated per month, per hour, or by customer usage:

- $1 per faxed page
- $19.95 per month for interactive games service
- 10 free hours per month for Internet service
- $.50 per email sent by the customer
- Purchase rates for products that customers can purchase in quantity

  For example, you might sell a product that provides 100 hours of service a month. If customers can purchase two of these products for a total of 200 hours, use the scaled impact for the purchase rate.

As a rule, when you do not know the exact amount that will be rated, specify scaled impact.

Even though monthly rates are consistent from one month to the next, you still use a scaled impact. This is because they charge an amount per event; in this case, a monthly event generated by BRM:

Scaled rate of $10 per month **X** event quantity of 1 month **=** balance impact of $10

About Fixed Balance Impacts

A *fixed amount* is a predefined amount that is always applied to the account when any amount of the event quantity is rated.

Use a fixed impact to define rates based on occurrence. These are amounts that do not change with usage or with the quantity of events being rated. For example:

- $5 installation fee
- $10 cancellation fee
- 10 hours of free service when you register for a service

An event that uses a fixed amount does not impact the account balance until the event ends. This enables BRM to calculate variable fees for the same event type. For example, when a single balance impact has both a fixed amount and a scaled amount, BRM can sum the impact before the account balance is affected.

Specifying Multiple Balance Impacts for a Single Rate

You can specify more than one balance impact per rate:

- For the same event, you can apply a balance impact to a currency resource and to a noncurrency resource. For example, you might create a product that charges $20 per month and gives the customer 10 free Internet access hours. In that case, the same rate would have two balance impacts: one for US dollars and one for Internet hours.
- For the same event, you can apply a balance impact based on quantity. For example:
  - For 0 to 10 faxes, charge $0.50 per fax
  - For 11 or more faxes, charge $0.10 per fax

  For more information, see "Real-Time Rating Based on Event or Resource Quantity".

Copyright © 2025, 2026, Oracle and/or its affiliates.

Specifying the Validity Period of Granted Resources

You specify validity periods for balance impacts that grant resources such as free minutes. The validity period defines when a granted resource is available for consumption by the subscriber's usage. The subscriber can consume the resource balance during the valid period only.

You can set the balance impact to start immediately, relative to when the resource is granted, or on first usage (when the subscriber consumes the resource for the first time). Setting resources to start on first usage is useful, for example, to provide free resources for a limited time, beginning when the subscriber starts consuming the resource. For more information, see "About Balance Impacts That Become Valid on First Usage".

Assigning Impact Categories to Balance Impacts

You use impact categories to apply different balance impacts when using the same rate plan. For example, to rate calls made to two different countries, you create impact categories for each country. When the call event occurs, the impact category determines which balance impacts to apply.

If you are using a rate plan selector, you assign impact categories to balance impacts when you create a rate. When the rate plan selector selects a rate plan and impact category, only those balance impacts with the matching impact category are applied. See "Real-Time Rating Using Event Attributes".

Assigning General Ledger (G/L) IDs to Balance Impacts

To collect information for your general ledger accounting system, you associate types of balance impacts with general ledger (G/L) IDs.

For example, when you create a usage rate, you can associate it with a G/L ID that records the revenue from all usage-rate balance impacts.

Allowing Charges to Be Sponsored

When charges are sponsored by a sponsor group, the sponsor group owner account receives the sponsored balance impacts generated by the group's member accounts. Therefore, you can use sponsor groups to let one customer pay part or all of other customers' service charges.

You define which accounts are sponsor group owners and which are members by creating sponsor groups in Customer Center. When you create a sponsor group, you select rate plans for the group to sponsor. You can select rate plans from all the products in your company's price list except the default product.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> Creating sponsor groups is easier if all resources in the rates in sponsored rate plans are sponsorable. You might want to name your products so they can be identified as products that contain sponsorable rate plans.
>
> If your company supports branded service management, this list displays all the products in your company's price list that are available for the sponsor group owner's brand. If your company does not support brands, this list displays all the products in the price list.

One Sponsor per Product

Each products in a member account should have only one sponsor. If group A sponsors purchase charges for a service and group B sponsors monthly subscription charges for the same service, the purchase and cycle rate plans should be in different products. If both rate plans are in the same product, BRM assigns all the product's charges to only *one* of the two sponsor groups. If the charges are assigned to group A, only the purchase fees for the service are sponsored. Members of group B must pay their own subscription fees.

You can also provide commissions by using service provider management. See "About Remittance".

About Balance Impacts That Become Valid on First Usage

You can set the validity period of resources granted by products and discounts to start on first usage: when subscribers begin consuming the resource balances for the first time.

For example, a deal grants 300 free minutes and 5 days of free video streaming. You set the free minutes to be valid immediately and set the streaming video to be valid when first used. A subscriber purchases the deal on June 1 and uses the video streaming service for the first time on June 5. The free minutes are valid starting June 1 and the streaming video is valid from June 5 to June 10.

You can also set the products and discounts that grant resources to start on first usage. There is no dependency between a product or discount's first usage start time and the first usage start time of the resource it grants. For information about products and discounts that start on first usage, see "About Effective Periods That Start on First Usage".

Resources whose validity period starts on first usage are added to the account balance at the time of the grant, but the validity period is not set until one of the following occurs:

- (Default) The first usage session in which the resource is consumed ends. The validity start time is set to the end time of the session.
- The resource is first authorized to be consumed. The validity start time is set to the start time of the first session.

The end time is set based on the end time that you configure for the resource and is set relative to the start time.

For more information, see "About Setting Resource Validity Periods Based on First Usage".

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> If the product granting the resource also starts on first usage, when the product is first used and its validity period is set, resources that are granted by that product cannot be consumed by the first usage event during real-time rating. This is because real-time rating does not generate the purchase and cycle events that grant resources until after the first usage event is rated. However, resources granted by products are available for consumption during real-time discounting because discounting occurs after the purchase and cycle events have been processed. For information about products and discounts that start on first usage, see "About Effective Periods That Start on First Usage".

If a product or discount is canceled after being used, the validity end time of any resources granted by the product or discount is set to the time of the cancellation.

About Setting Resource Validity Periods Based on First Usage

> **Note**
>
> This section does not apply to products and discounts whose validity periods start on first usage. For information about them, see "About Effective Periods That Start on First Usage".

To set a resource validity period based on first usage, BRM must first set the period's start time. The end time is then set relative to the start time.

A resource validity period's start time is set during one of the following phases of the first usage session:

- **Stop Accounting:** (Default) The period's start time is set to the end time of the first usage session. This is recommended for typical validity periods.
- **Authorization:** The period's start time is set to the start time of the first usage session. This is recommended for short validity periods, such as one-day promotions. It prevents users from extending the promotional period by leaving the usage session open and thus preventing BRM from setting the validity period's end time.

  > **Note**
  >
  > If usage does not occur after the validity period's start time is set during the authorization phase (for example, a phone call is authorized but not answered), the validity period's start time nonetheless remains in effect and cannot be reset.

To specify when the start time is set, use the **SetFirstUsageInSession** business parameter. See "Setting Validity Start Time to the Start Time of the First Usage Session".

Setting Validity Start Time to the Start Time of the First Usage Session

By default, start times for resource validity periods based on first usage are set to the end time of the first usage session. Alternatively, they can be set to the start time of the session. You

Copyright © 2025, 2026, Oracle and/or its affiliates.

change the time they are set to by using the **pin_bus_params** utility. For information about that utility, see "pin_bus_params" in *BRM Developer's Guide*.

To set validity start time to the start time of the first usage session:

1. Go to the *BRM_home*/**sys/data/config** directory, where *BRM_home* is the directory in which BRM is installed.
2. Run the following command, which creates an editable XML file from the **activity** instance of the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsActivity bus_params_act.xml
   ```

   This command creates the **bus_params_act.xml.out** file in your working directory. To place this file in a different directory, include the path in the file name.
3. Open the **bus_params_act.xml.out** file.
4. Search for the following line:

   ```
   <SetFirstUsageInSession>disabled</SetFirstUsageInSession>
   ```
5. Change **disabled** to **enabled**.
6. Save the file as **bus_params_act.xml**.
7. Go to the *BRM_home*/**sys/data/config** directory, which includes support files used by the **pin_bus_params** utility.
8. Run the following command, which loads this change into the **/config/business_params** object:

   ```
   pin_bus_params PathToWorkingDirectory/bus_params_act.xml
   ```

   where *PathToWorkingDirectory* is the directory in which **bus_params_act.xml** resides.

   > ⚠ **Caution**
   >
   > BRM uses the XML in this file to overwrite the existing **act** instance of the **/config/business_params** object. If you delete or modify any other parameters in the file, those changes affect the associated aspects of the BRM activity configuration.

9. Read the object with the **testnap** utility or Object Browser to verify that all fields are correct.
10. Stop and restart the Connection Manager (CM).

About First-Usage Start Time for Shared Resources

If a resource is shared among accounts in a resource sharing group, the resource validity period is set when any account in the group first impacts the resource balance. Because the same balance is shared with all accounts in the group, the validity period of that resource applies to all accounts.

About Synchronizing First-Usage Validity of Resources in Deals

You can synchronize the validity periods of all resources granted in a deal that have first-usage start times. This sets the validity of all granted resources with first-usage start times when one of those granted resources is first consumed.

Copyright © 2025, 2026, Oracle and/or its affiliates.

For example, a deal grants free minutes and free text messages and both are set to start on first usage. A subscriber purchases the deal and on June 5 uses the free minutes by making a phone call. When the call is rated, the validity periods for both free minutes and text messages are set to start on June 5.

Only first-usage resources in the deal that is used to rate the event are synchronized. The validity periods of other resources and first-usage resources granted by other deals are not changed.

If a resource with a first-usage start time is granted after the validity period of other first-usage resources has already been set, the validity of the newly granted resource is synchronized with the existing validity periods.

To synchronize first-usage resources, you select **Align deal resource validity on first usage** in the **Deal Attributes** tab of Pricing Center.

About Restricting Resource Validity End Time to the Product or Discount End Time

For resources that start on first usage, you can restrict the resource validity end times so they do not extend beyond the validity end time of the products or discounts that grant the resources. This ensures that the resource balance cannot continue to be consumed after the product or discount expires.

> **Note**
>
> When a product or discount is *canceled*, the validity period end time of resources granted by that product or discount is set to the time of the cancellation.

About Fold Events

A fold event is a special internal event that usually occurs at the end of the accounting cycle. A fold event can be configured to either change a resource balance (currency or noncurrency) to zero or convert the resource balance into other resources.

For example, you can create a fold event to cancel unused free hours at the end of each month. You can also create a fold event to credit frequent flyer miles for every $500 a customer spends on a service.

Fold events are triggered at the following times:

- During regular billing
- At product cancellation
- When Bill Now is invoked

When a fold event occurs, BRM checks whether the resource sub-balance or bucket is valid or expired. The resource is folded if the sub-balance is expired or is always valid.

For example:

- If 500 free minutes are valid forever; the resource can be folded at any time.
- If 500 free minutes expire on February 1, it can be folded after January 31 23:59:59. The end time of the fold event must be January 31 23:59:59 or later.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> If you use delayed billing, folds occur at the end of the delayed billing period. This is to ensure that delayed events can continue to consume granted resources before final billing is run. However, free resources are granted at the beginning of each cycle, before the fold event occurs. If the newly granted resources do not have a configured validity period (that is, their validity period never ends), the fold event will remove those resources.
>
> When you use fold events to remove unused resources and you also use delayed billing, you should always configure a validity period for the granted resources. This prevents fold events from removing resources that are granted at the beginning of a cycle before final billing is run.

You can configure fold events to convert resource R1 to R2. You can also convert R2 to R3 if resource R2 is available or tracked in the account's balance group *before* the fold event is triggered.

For example, a price plan has a cycle forward event that charges a $100 cycle fee and grants 1000 free minutes. When you create an account with this plan, the account's balance group contains two resources: USD and Free Minutes. If you configure a fold event to:

1. Convert the free minutes (R1) to free games (R2)
2. Convert the free games (R2) to Euro (R3)
3. Convert the free minutes (R1) balance to zero

The free games resource (R2) is not converted to R3 in step 2 because it is not available in the account's balance group when the fold is triggered, even though Free Games is added to the account's balance group in step 1.

To fold free games to Euro, you add Free Games to the plan in the **Track Balances** tab in the Plan Attributes window. When the account is created, Free Games is added to the account's balance group and its balance is set to zero (0).

By default, an account's resource balances are folded in ascending order based on the resource ID. You can change this order by modifying the policy opcode, PCM_OP_SUBSCRIPTION_POL_SPEC_FOLD.

You can create a fold to charge for overuse of free resources before the customer cancels a product.

Ensuring That All of an Event Is Rated

BRM cannot apply a rate if the credit limit for its resource has been reached. However, there might be times when an event, or part of an event, has not been rated, and all credit limits have been reached. To charge for the unrated event or event portion, you create a *limit override rate*. A limit override rate is used even though a credit limit has been reached.

In effect, a limit override rate sets the baseline charge for an event. When no other rates apply because of credit limits, you charge a default amount for the event.

Copyright © 2025, 2026, Oracle and/or its affiliates.

7

About Pipeline Rating

This chapter provides an overview of how to use the Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager to rate usage events.

Topics in this document:

- About Batch Rating
- How Events Are Rated by Using Pipeline Manager
- How an EDR Is Processed in a Pipeline
- About the Oracle CDR Format
- About EDRs
- About Mapping EDR Field Names and Alias Names
- About Function Modules
- About iScripts and iRules
- How Pipeline Manager Uses BRM Data
- About Loading Pipeline-Rated Event Data
- About Pipeline Rating and BRM Billing
- Function Module Dependencies
- Data Module Dependencies

About Batch Rating

Batch rating is typically used for rating telephony services, where large amounts of events are recorded in files. Batch rating is usually performed by Pipeline Manager, but you can also perform batch rating by using Universal Event (UE) Loader.

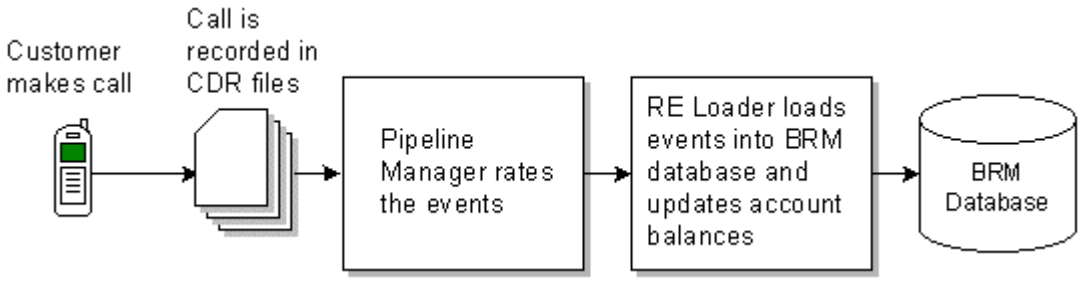
For example, to rate telephone calls:

1. Pipeline Manager reads the data from CDRs to determine the time and duration of the calls.
2. Pipeline Manager rating modules apply a charge to the amount of time. For example, if you specify a charge of 10 cents per minute, a 10-minute call costs $1. This charge is called the *balance impact*.
3. Rated Event (RE) Loader loads the rated events into the BRM database.
4. BRM adds the total charge for the event to the customer's account balance.

Figure 7-1 shows how a billable event is rated by Pipeline Manager and recorded in the BRM database:

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 7-1 Pipeline Manager Event Rating and Recording to BRM Database



How Events Are Rated by Using Pipeline Manager

Pipeline Manager rates only usage events, not recurring events, and not one-time events. Pipeline Manager rates usage events as follows:

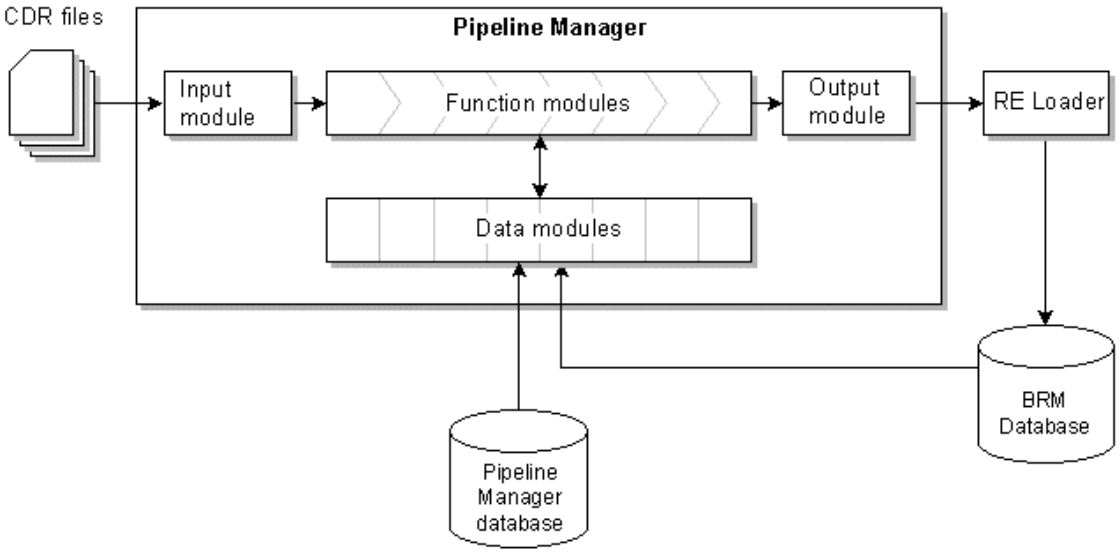
1. Call detail record (CDR) files are collected from network switches and processed by mediation software. The mediation software converts data into the Oracle CDR format so it can be read by a pipeline input module. See "About the Oracle CDR Format".

   Mediation can also remove records that are not needed for rating. For example, it can remove records for free text messages reminding customers of a missed call.
2. The mediation software places the CDR file in a predefined directory.
3. The pipeline input module reads the CDR file and begins processing it. The input module does the following:
   - Performs error checking on the input CDR.
   - Converts the data in the CDR file into event data records (EDRs). To do so, the input module normalizes the raw data that represents each event and formats it into a standard structure that can be processed by the pipeline modules.
4. Function modules, working with data modules, rate the events.
   - *Function modules* perform the preprocessing and rating operations. For example, they check for duplicated EDRs, determine the quantity to rate, the zone to apply, and the charge for the event.
   - *Data modules* supply data to the function modules by reading from the Pipeline Manager database or the BRM database.
5. The output module writes data to an output file.
6. Rated Event (RE) Loader reads the output file, loads the events into the BRM database, and updates the customer's account balance.

Figure 7-2 shows how usage events are rated:

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 7-2 Usage Events Rating



When you configure Pipeline Manager, you do the following:

- Configure the input module to accept and process your type of CDR file.
- Configure function and data modules to carry out normalization, zone mapping, and rating.
- Configure the output module to output the data.

In addition to configuring Pipeline Manager, you need to do the following:

- Configure the Oracle Data Manager (DM) to transfer data from the BRM database to Pipeline Manager.
- Configure RE Loader to load rated events into the BRM database. See "About Loading Rated Events into the BRM Database" in *BRM Loading Rated Events*.

How an EDR Is Processed in a Pipeline

When an EDR is sent through a pipeline, function modules perform the following types of operations:

- The input module processes the CDR and creates an event data record (EDR) for each event. Processing the CDR file includes:
  - Identifying each type of record in the file; for example, header records, event records, and trailer records.
  - Normalizing data and translating it into the internal EDR format.
- Preprocessing modules prepare EDRs for rating. For example:
  - The FCT_DuplicateCheck module discards duplicate EDRs.
  - The FCT_CallAssembling module assembles calls that have been split into multiple records.
  - The FCT_Reject module rejects EDRs with errors.

  Preprocessing modules typically run at the beginning of a pipeline.
- Enrichment modules normalize or add data that the rating modules need. For example:

Copyright © 2025, 2026, Oracle and/or its affiliates.

  - The FCT_ServiceCodeMap module assigns an internal service code to identify which service generated the event.
  - The FCT_CliMapping module maps multiple phone numbers to a single phone number, so customers can be billed for all of their phones on one bill.

  Enrichment modules typically run before rating modules in a pipeline.
- Zoning modules calculate geographic or area-code-based zones for rating purposes.
- Rating modules perform rating.
- Discount modules perform discounting, after the EDR has been rated.
- The aggregation module collects data for reports.

About the Order of Modules in a Pipeline

Some modules need to be configured in a specific order. For example:

- You use the FCT_Discard module to discard EDRs that you don't want to rate. You need to discard them before the rating modules process them; otherwise you spend system resources on rating unwanted EDRs.
- To provide discounts, the discount module needs to work with events that have already been processed by the rating modules.

For information about the order of modules in a pipeline, see "Function Module Dependencies".

About the Oracle CDR Format

The Oracle CDR format is the standard CDR file format used by Pipeline Manager for processing CDRs. For example, the Oracle CDR format is used by pipelines to generate CDRs that are passed between pipelines for additional processing, such as between a preprocessing pipeline and a rating pipeline.

The Oracle CDR format can also be used as the input format for rating pipelines. Additionally, pipelines can be configured to translate other formats into the Oracle CDR format for rating.

For complete details about the Oracle CDR format structure, see "BRM Rating EDR Container Description" in *BRM Pipeline Manager Reference*.

For information about the input and output processes, see "Configuring EDR Input Processing" and "Configuring EDR Output Processing" in *BRM Setting Up Pipeline Rating and Discounting*.

About EDRs

The data for each event is stored as an EDR. As an EDR is processed, function modules process data in it or add data. For example, the FCT_CustomerRating module adds the charge code. The FCT_MainRating module uses that code to calculate and add the charge amount.

The following sample shows a portion of a charge packet in an EDR. Each field stores a specific piece of data; for example, the RATEPLAN_CODE field stores the charge used for rating the event.

```
CHARGE_PACKET
RATEPLAN_CODE:              <SIMPLERATE>
RATEPLAN_TYPE:              <R>
ZONEMODEL_CODE:             <MT_ZM>
SERVICE_CODE_USED:          <SMS>
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
SERVICE_CLASS_USED:          <DEF>
IMPACT_CATEGORY:             <MT_SMS>
TIMEMODEL_CODE:              <TM_MTEL>
TIMEZONE_CODE:               <TZ_WKNDDT>
DAY_CODE:                    <WEEKEND>
TIME_INTERVAL_CODE:          <0800_2000>
PRICEMODEL_CODE:             <MT_SMS>
PRICEMODEL_TYPE:             <S>
RESOURCE:                    <YEN>
RUMGROUP:                    <EVENT>
RUM:                         <EVT>
CHARGE_TYPE:                 <N>
CHARGING_START_TIMESTAMP:    <20020901182200>
CHARGEABLE_QUANTITY_VALUE:   <1>
CHARGED_CURRENCY_TYPE:       <R>
CHARGED_AMOUNT_VALUE:        <50>
CHARGED_AMOUNT_CURRENCY:     <JPY>
CHARGED_TAX_TREATMENT:       <N>
CHARGED_TAX_RATE:            <0>
CHARGED_TAX_CODE:            <NORM>
USAGE_GL_ACCOUNT_CODE:       <50000>
```

About EDR Containers

EDR containers are temporary in-memory data structures for transporting EDRs through function modules. Each container stores data for a header record, a detail record, or a trailer record.

When a transaction starts:

1. The EDR Factory creates an EDR container according to the information in the container description file.
2. The input module writes data to relevant fields in the container. For more information, see "Configuring EDR Input Processing" in *BRM Setting Up Pipeline Rating and Discounting*.
3. The function modules process data in or add data to particular fields. For example, the FCT_BillingRecord function module processes balance impact-related data.
4. The EDR Factory empties the container and releases the cache, which will be used for other containers.

About the EDR Container Description

The data in an EDR, the default values, and the way the data is organized are defined in a *container description*. A typical container description defines the following types of containers:

- **A header container type.** The header container type contains information stored in a header record; for example, the country of origin, originating network, and creation time. It also includes sequence numbers for ensuring that EDRs are processed correctly. A separate EDR is created for the header record.
- **One or more basic detail container types.** A basic detail container includes the data in an event record; for example, a phone call. This record includes information such as the A number and the service that generated the event. An EDR is created for each detail record. Each EDR can contain data for one service only; for example, GSM or GPRS.

  Each detail record includes one or more associated records. These records include service-specific data, zoning data, and rating data. See "About Associated Records".

Copyright © 2025, 2026, Oracle and/or its affiliates.

- **A trailer container type.** The trailer container type contains information stored in a trailer record; for example, the number of records. A separate EDR is created for the trailer record.

In almost all cases, you can use the BRM EDR container description. You can also customize the data that is included in an EDR by customizing the container description.

> **Note**
>
> If you customize the container description, you need to make sure that your customizations do not affect existing module functionality. For example, many modules require data from a specific EDR field.

About the Container Description File

The container description file is an ASCII file that defines how to build EDR containers. You use the file to define the data in the EDR container, the default values, and the way the data is organized. For more information, see "About the EDR Container Description".

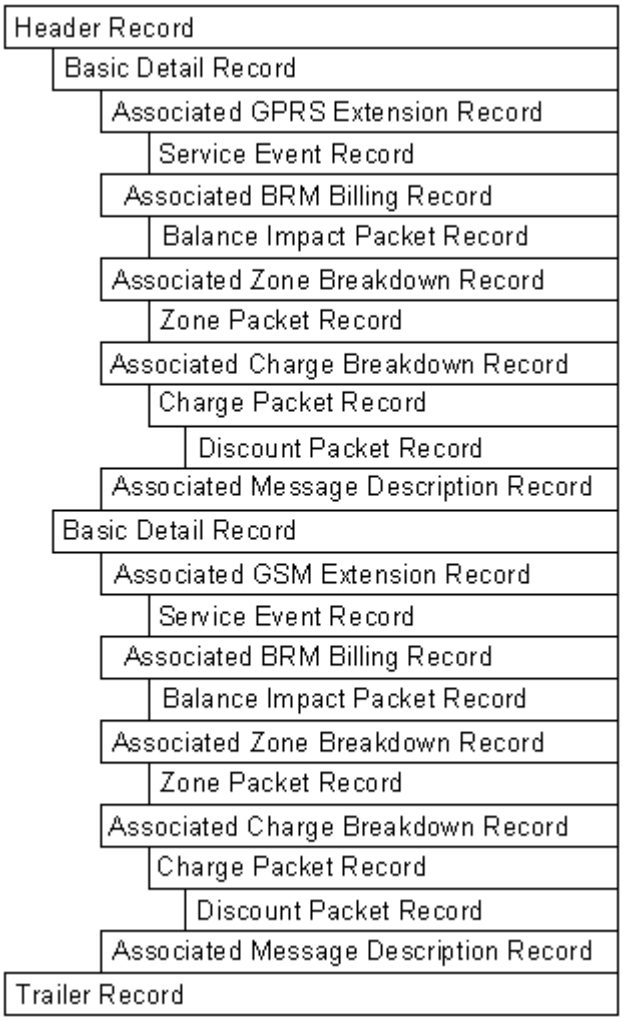
The default container description file is *Pipeline_home*/**formatDesc/Portal/containerDesc.dsc**, where *Pipeline_home* is the directory in which you installed Pipeline Manager. This file covers the needs of most input file formats, but you can customize it to meet your business requirements.

> **Note**
>
> If you alter or add fields to the container description file, you may also need to write custom iScripts to access these new fields.

Figure 7-3 shows the BRM EDR container organization:

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 7-3 BRM EDR Container Organization



About Associated Records

Associated records are added to EDRs by different modules; for example, the FCT_SegZoneNoCust module adds an associated zone breakdown record.

- **Associated service extension record**. This record stores specific information about the service that generated the event; for example, the originating switch and the device number.

  EDRs for GSM events can include one or more service event records. This record includes event information, such as equipment used and details about supplementary services.

- **Associated BRM billing record**. This record stores rated event data that is loaded into the BRM database. It includes the POID of the **/account** object and **/service** object and the POID of the item that receives the balance impact. This record is created by the FCT_BillingRecord module.

  If an event affects more than one customer balance, an associated BRM billing record is created for each balance.

Copyright © 2025, 2026, Oracle and/or its affiliates.

An associated BRM billing record can contain one or more *balance impact packets*. These contain data about the event. Each balance impact packet includes data for one balance impact per balance element, and, optionally, per G/L ID.

- **Associated charge breakdown record**. These records hold the charges for an event. For example, the FCT_CustomerRating module adds an associated charge breakdown record to the EDR to record the rating results.

  Each associated charge breakdown record includes one or more charge packets. Each charge packet includes a single charge; for example, the charge for a telephone call for a single time period.

  Each charge packet can include one or more *discount packets*. A discount packet includes information about the discount owner and rollover information.

  During rating, Pipeline Manager might generate several charge packets. For example:

  – When rating multiple balance elements, each balance element has its own charge packet.

  – When using multiple ratable usage metrics (RUMs), each RUM has its own charge packet.

  – When splitting charges across time zones, each time zone has its own charge packet.

  – If multiple currencies are used, each currency has its own charge packet.

- **Associated zone breakdown record**. This record contains data for zoning. It is created by the FCT_SegZoneNoCust module. This is used for comparative analysis of different zoning options.

  An associated zone breakdown record includes one or more *zone packet records*. Each of these records includes data about a single zone model.

- **Associated message description record**. This record holds information and error messages.

How an Input File Is Represented in EDR Containers

A typical conversion from an input file to EDR containers creates the following EDRs:

- A set of control EDRs that the pipeline uses for managing transactions. For example, this control EDR specifies to start a transaction:

```
= = = = =  S T A R T   O F   C O N T A I N E R  = = = = =
Container type = <SERVICE>
Content type   = <BEGIN_TRANSACTION>
Originator     = <ifw.Pipelines.W_SAMPLE.Input>
Stream Number  = <0>
IsValidDetail  = <false>
Record number  = <1>
has Errors     = <0>
.
.
.
= = = = = =  E N D   O F   C O N T A I N E R  = = = = = =
```

- A header record that includes information about the input file:

```
= = = = =  S T A R T   O F   C O N T A I N E R  = = = = =
Container type = <DATA>
Content type   = <HEADER>
Originator     = <>
Stream Number  = <3>
IsValidDetail  = <false>
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
Record number  = <0>
has Errors     = <0>

HEADER
------

RECORD_LENGTH:                    <0>
RECORD_TYPE:                      <>
RECORD_NUMBER:                    <0>
.
.
.
= = = = = =  E N D   O F   C O N T A I N E R  = = = = = =
```

- Multiple detail records, each containing the data for one event:

```
= = = = =  S T A R T   O F   C O N T A I N E R  = = = = =
Container type = <DATA>
Content type   = <DETAIL>
Originator     = <>
Stream Number  = <3>
IsValidDetail  = <true>
Record number  = <1>
has Errors     = <0>

DETAIL
------

RECORD_LENGTH:                    <0>
RECORD_TYPE:                      <020>
RECORD_NUMBER:                    <0>
DISCARDING:                       <0>
TYPE_OF_A_IDENTIFICATION:         <S>
A_MODIFICATION_INDICATOR:         <00>
A_TYPE_OF_NUMBER:                 <0>
A_NUMBERING_PLAN:                 <0>
A_NUMBER:                         <008190115551212>
.
.
.
= = = = = =  E N D   O F   C O N T A I N E R  = = = = = =
```

- A trailer record, which includes information about the records in the file:

```
= = = = =  S T A R T   O F   C O N T A I N E R  = = = = =
Container type = <DATA>
Content type   = <TRAILER>
Originator     = <>
Stream Number  = <3>
IsValidDetail  = <false>
Record number  = <7>
has Errors     = <0>

TRAILER
-------

RECORD_LENGTH:                    <0>
RECORD_TYPE:                      <090>
RECORD_NUMBER:                    <14>
.
.
.
= = = = = =  E N D   O F   C O N T A I N E R  = = = = = =
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

- A set of control EDRs. This EDR specifies the end of a transaction:

```
= = = = =  S T A R T   O F   C O N T A I N E R  = = = = =
Container type = <SERVICE>
Content type   = <END_TRANSACTION>
Originator     = <ifw.Pipelines.W_SAMPLE.Input>
Stream Number  = <0>
IsValidDetail  = <false>
Record number  = <8>
has Errors     = <0>
.
.
.
= = = = = =  E N D   O F   C O N T A I N E R  = = = = = =
```

How EDRs Are Used for Managing Transactions

Transactions are managed by using the EDR content type:

- When an EDR with the BEGIN_TRANSACTION content type is processed, a new transaction is started. This usually occurs at the beginning of each input file.
- When an EDR with the END_TRANSACTION content type is processed, the transaction is ended. This usually occurs at the end of each input file.
- When an EDR with the STOP content type is processed, this indicates that the pipeline is shutting down gracefully and gives the modules a chance to save their state and prepare for the shutdown.

About Mapping EDR Field Names and Alias Names

The EDR container description defines the data in the EDRs and the name of each field. See "About the EDR Container Description".

In the pipeline modules, EDR fields can optionally be represented using two names:

- The EDR container field name; for example, DETAIL.RECORD_LENGTH. This is the input format of the field.
- An alias name, to which the container field name is mapped; for example, BDR_RECORD_LENGTH.

The sample below shows EDR container field names on the left and internal alias names on the right:

```
DETAIL.RECORD_LENGTH     ->  BDR_RECORD_LENGTH
DETAIL.RECORD_TYPE       ->  BDR_RECORD_TYPE
DETAIL.RECORD_NUMBER     ->  BDR_RECORD_NUMBER
DETAIL.DISCARDING        ->  DISCARDING
DETAIL.CHAIN_REFERENCE   ->  CHAIN_REFERENCE
```

Pipeline Manager function modules and iScripts use the alias name for manipulating data. When you write a custom iRule or an iScript, you must create an alias name for the EDR container fields that your module uses. Using alias names facilitates customization because you can use the container fields with different names and hierarchies without the need to change the module source code. See "Viewing and Creating Alias Mapping for an EDR Field".

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> The same EDR container field can be mapped to different alias names in different modules.

The list of BRM-defined alias names is in *Pipeline_home***/formatDesc/Portal/AliasFieldList.dsc**.

Viewing and Creating Alias Mapping for an EDR Field

Use Pricing Center or Pipeline Configuration Center (PCC) to view a list of existing alias mappings and to add custom alias mappings.

The Alias Mapping table lists the existing alias names including the BRM-defined alias names. Each entry includes the following mapping information:

- **EDR Container Description**, which specifies the EDR container to which the EDR field belongs.
- **Reference**, which is the pipeline module that uses the alias name. Its value is the pipeline module reference in the module block of the pipeline registry file; for example, **PipelineSplit** in the following registry entry:

```
PipelineSplit
{
   ModuleName = FCT_IRules
   Module
  {
    …
   }
}
```

> **Note**
>
> If the value of **Reference** is Account_CustA, Account_CustB, UniData_CustA, or UniData_CustB, the alias mapping table entry defines the EDR field from which to get the account identifier. If the value is RUM, the alias mapping defines the EDR field from which to get the rateable usage metrics (RUM) quantity. If the value is UOM, the alias mapping defines the EDR field from which to get the unit of measure for a RUM

- **Key**, which is the alias name for the EDR container field.

> **Note**
>
> For Account_CustA, Account_CustB, UniData_CustA, and UniData_CustB, this is the service code.

- **Type**, which is **Internal** for the alias mappings used by the default pipeline modules and **Plugin** for the modules used by custom modules.
- **Field ID**, which is the name of the field in the EDR container.
- **History**, which specifies when the alias mapping was created or modified.

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Function Modules

Function modules perform all the rating tasks in a pipeline. In addition, they perform EDR management tasks, such as ensuring that duplicate EDRs aren't processed.

There are different categories of function modules. They generally run in the following order:

- Preprocessing modules prepare EDRs for rating. See "About Preprocessing Modules".
- Enrichment modules add data to EDRs. See "About Enrichment Modules".
- Service mapping modules map external service codes to internal service codes. See "About Service Mapping Modules".
- Zoning modules calculate zone data to use for rating. See "About Zoning Modules".
- Rating modules rate the events. See "About Rating Modules".
- Discount modules adjust the charges. See "About Discounting Modules".
- Roaming modules are specialized zoning and rating modules that handle roaming events. See "About Roaming Modules".

About Preprocessing Modules

Use the preprocessing modules to handle the following tasks:

- Use the "FCT_CallAssembling" in *BRM Pipeline Manager Reference* module to assemble calls that have been split into multiple EDRs. See "Assembling EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.
- Use the "FCT_Reject", "FCT_PreSuspense", and "FCT_Suspense" modules in *BRM Pipeline Manager Reference* to handle EDRs that contain errors. Rejected EDRs are sent to a separate output stream. You can then fix the problem that caused them to be rejected and re-process them. See in *BRM Suspending and Recycling Pipeline EDRs* for more information.
- Use the "FCT_DuplicateCheck" module in *BRM Pipeline Manager Reference* to discard duplicate EDRs. See "Handling Duplicate EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.
- Use the "FCT_Discard" module in *BRM Pipeline Manager Reference* to discard EDRs that you don't want to process. See "Discarding and Skipping EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.
- Use the "FCT_EnhancedSplitting" module in *BRM Pipeline Manager Reference* to send EDRs to different output streams based on rules that you define. For example, you can split EDRs from roaming outcollects and incollects into different streams. See "Using Rules to Send EDRs to Different Output Streams" in *BRM Setting Up Pipeline Rating and Discounting*.
- Use the "IRL_EventTypeSplitting" iRule in *BRM Pipeline Manager Reference* to send EDRs to different output streams based on the service. See "Sending EDRs to Pipeline Output Streams" in *BRM Setting Up Pipeline Rating and Discounting*.
- Use the "FCT_AccountRouter" in *BRM Pipeline Manager Reference* to send EDRs to the correct pipeline in a multischema system. See "Using Pipeline Manager with Multiple Database Schemas" in *BRM Setting Up Pipeline Rating and Discounting*.

Copyright © 2025, 2026, Oracle and/or its affiliates.

You set up EDR preprocessing by configuring modules in a pipeline. For information on configuring preprocessing modules, see "Configuring EDR Preprocessing" in *BRM Setting Up Pipeline Rating and Discounting*.

About Enrichment Modules

When you enrich EDR data, you add or change data for further processing.

- Use the "FCT_CliMapping" module in *BRM Pipeline Manager Reference* to specify that multiple phone numbers owned by one customer are charged for on one bill. See "Mapping Multiple Phone Numbers to a Single Number" in *BRM Setting Up Pipeline Rating and Discounting*.
- Use the "FCT_SocialNo" module in *BRM Pipeline Manager Reference* to identify "social numbers" that are not displayed on an invoice. See "Setting Up Social Numbers" in *BRM Setting Up Pipeline Rating and Discounting*.
- Use the "FCT_NOSP" module in *BRM Pipeline Manager Reference* to identify a network operator and service provider when performing segment rating. See "Identifying the Network Operator/Service Provider" in *BRM Setting Up Pipeline Rating and Discounting*.
- Use the "FCT_NumberPortability" module in *BRM Pipeline Manager Reference* to process events correctly when a customer changes network provider but keeps the phone number. See "Setting Up Number Portability" in *BRM Setting Up Pipeline Rating and Discounting*.
- Use the "FCT_PrefixDesc" module in *BRM Pipeline Manager Reference* to specify how to identify the destination of a call based on the number prefix. See "Creating Call Destination Descriptions" in *BRM Setting Up Pipeline Rating and Discounting*.

You set up EDR enrichment by configuring modules in a pipeline. For more information about enrichment modules, see "Setting Up EDR Enrichment" in *BRM Setting Up Pipeline Rating and Discounting*.

About Service Mapping Modules

Incoming EDRs from multiple switches often use different codes to represent the same service or supplementary service. To process EDRs, you need to normalize that data by mapping external service codes to internal service codes, service classes, and usage classes. You use the following modules:

- Use the "FCT_ServiceCodeMap" module in *BRM Pipeline Manager Reference* to map external service codes to internal service codes. For example, CDRs might use different codes for different types of telephony services. You can map those codes to a single code; for example, TEL. The service is typically a bearer or primary service.
- Use the "FCT_UsageClassMap" module in *BRM Pipeline Manager Reference* to map external codes for supplementary services, such as call forwarding, to internal usage classes. Usage classes are typically used for rating based on quality of service, sub-services, or service-level agreement values.
- Use the "IRL_UsageType" iScript in *BRM Pipeline Manager Reference* to assign usage types to EDRs. A usage type is an internal code that represents an attribute of a customer account; for example, a friends and family discount.
- Use the "FCT_UoM_Map" module in *BRM Pipeline Manager Reference* to convert the unit of measurement (UoM) of an incoming EDR to a UoM needed for rating a particular service. For example, an EDR might include the usage amount in seconds, but the service is configured to charge by minutes. The FCT_UoM_Map module converts seconds to minutes, using rounding rules.

Copyright © 2025, 2026, Oracle and/or its affiliates.

You configure service mapping by configuring mapping modules in a pipeline.

About Zoning Modules

Zoning function modules prepare EDRs for rating by identifying geographic or logical zones that are used for rating.

You can identify zones as follows:

- Using logical source and destination; for example, area codes, the service used, or retail or wholesale usage.
- Using geographic distance.

Zoning is performed by the following function modules:

- Use the "FCT_PreRating" module in *BRM Pipeline Manager Reference* to calculate zones and impact categories.
- Use the "FCT_APN_Map" module in *BRM Pipeline Manager Reference* to process APN data for zoning.
- Use the "FCT_USC_Map" module in *BRM Pipeline Manager Reference* to refine impact categories based on service attributes.
- Use the "FCT_Zone" module in *BRM Pipeline Manager Reference* to compute zones when you use a pipeline only for zoning.

About Rating Modules

Rating modules rate EDRs. Each module performs a specific function; for example, the FCT_Dayrate module determines the time of day used for rating. Because each module adds or modifies data, the order of modules is important. For example, the FCT_RateAdjust module must process EDRs after the FCT_MainRating module. For more information, see "Function Module Dependencies".

- Use the "FCT_GlobalRating" module in *BRM Pipeline Manager Reference* to apply a global charge to all EDRs.
- Use the "FCT_CustomerRating" module in *BRM Pipeline Manager Reference* to provide customer data for other rating modules.
- Use the "FCT_RSC_Map" module in *BRM Pipeline Manager Reference* to perform rating based on the quality of service when you set up service-level agreements (SLAs).
- Use the "FCT_RateAdjust" module in *BRM Pipeline Manager Reference* to adjust charges after rating.
- Use the "FCT_BillingRecord" module in *BRM Pipeline Manager Reference* to consolidate billing information for loading into the BRM database.

About Discounting Modules

Discounting modules run after rating modules. Pipeline discounting uses the following processes:

- The module checks for the conditions that allow a discount; for example, using 100 minutes per month.
- The module grants the discount; for example, reducing the charge by 10%.

You set up discounting by configuring the following discounting modules in a pipeline:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Use the "FCT_DiscountAnalysis" module in *BRM Pipeline Manager Reference* to analyze EDRs before discounting.
- Use the "FCT_Discount" module in *BRM Pipeline Manager Reference* to calculate and apply discounts.

About Roaming Modules

Roaming modules rate roaming usage. You set up roaming rates by configuring the following modules in a pipeline. Use the "FCT_CarrierIcRating" module in *BRM Pipeline Manager Reference* to supply the interconnect charge for the FCT_MainRating module.

About iScripts and iRules

iScripts and iRules perform processing tasks similar to function modules. They are run by the FCT_iScript and FCT_iRules modules. In addition to the iScripts and iRules provided by BRM, you can create your own iScripts and iRules.

How Pipeline Manager Uses BRM Data

Pipeline Manager needs to get data from the BRM database to rate each account. For example, Pipeline Manager gets the charge to use from the services and products/ charge offers owned by an account. Pipeline Manager also gets historical data; for example, if a customer changes a phone number, Pipeline Manager needs the old number to rate calls made using it.

Pipeline Manager also needs to get data that is not required for rating but is required by the BRM event objects that Rated Event (RE) Loader loads into the BRM database. For example, every event requires an item, so Pipeline Manager needs to get the correct item for the event. When the rated event is loaded, the correct item is already recorded in the event.

How Pipeline Manager Identifies Accounts

When Pipeline Manager rates usage events, there is no information in the original CDR that specifies which BRM account was responsible for the event. Pipeline Manager needs to know the account to rate the event and to apply discounts.

To find the account:

1. In the EDR, Pipeline Manager finds the phone number identified as the calling number.
2. In the data retrieved from BRM, this number is stored in the alias list in a service object. Once the service object is found, it has a pointer to the account object, from which Pipeline Manager identifies the account.

How Pipeline Manager Chooses a Charge

Pipeline Manager needs a charge for determining usage charges. There are three ways that Pipeline Manager can find out which charge to use:

- **Product/Charge offer priority**. By default, Pipeline Manager uses the charge associated with the highest-priority product/charge offer. That is, it searches through all purchased products/charge offers, from highest priority to lowest priority, until it finds one that matches the event's rating criteria, such as the zone model and time model type. It then selects the charge associated with that charge offer. You assign priorities to products/charge offers.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- **Lowest charge**. When configured for least cost rating, Pipeline Manager finds the product/charge offer that generates the lowest overall charge to the customer and then uses the charge associated with the product/charge offer. You configure a pipeline for least cost rating by using the IRL_LeastCostPerEDR and ISC_LeastCost modules. See "About Least Cost Rating" in *BRM Setting Up Pipeline Rating and Discounting*.
- **ERA**. If a service or an account is associated with an extended rating attribute (ERA), Pipeline Manager uses the charge you configure and prioritize for the ERA.

> **Note**
>
> If a subscription service and member service both own a service-level ERA, the member service's ERA has priority and is used for selecting the charge.

How Pipeline Manager Assigns Delayed Events to Items

When Pipeline Manager outputs events to RE Loader, the events must include all mandatory event data. Most of that data comes from the incoming CDR; for example, the call origin and destination. Some of the data must come from BRM, including which bill item stores the balance impact of the event.

In BRM, every event object is associated with a bill item. The incoming CDR does not have any information about bill items, so Pipeline Manager gets that information from BRM.

1. In the EDR, Pipeline Manager finds the phone number identified as the calling number.
2. In the data retrieved from BRM, this number is stored in the alias list in a service object. Once the service object is found, Pipeline Manager uses the information in the item POID list to determine which bill item the event applies to.

BRM creates service usage items for the next accounting cycle on four occasions:

- When the service is purchased by an account.
- When billing is run.
- When the Bill Now feature is used in Billing Care.
- When a usage event occurs.

However, BRM does not create a usage item when Pipeline Manager processes an event. If no usage item exists, Pipeline Manager rejects the event. Therefore, BRM pre-creates usage items for Pipeline Manager to use.

BRM pre-creates items in the following cases:

- When an account is created.
- When you run billing.
- When a CSR uses Bill Now.
- At the end of the accounting cycle when you use delayed billing.

> **Note**
>
> To maintain correct discount balances, and to minimize rejected EDRs, always use delayed billing when you use pipeline rating.

Copyright © 2025, 2026, Oracle and/or its affiliates.

To choose the correct bill item, Pipeline Manager can do one of the following:

- Assign the event to the current open bill item.
- Assign the event to the next open bill item.
- Reject the event because it belongs to an item that has already been included in a bill.

To decide which item to apply the bill to, Pipeline Manager takes into account the following dates:

- The date when the call occurred (the EDR date).
- The current system date.
- The date when the current accounting cycle ends. This is called the *next accounting cycle* date.
- The number of days after the current accounting cycle ends when delayed billing runs. This number is called the *delayed billing offset*.

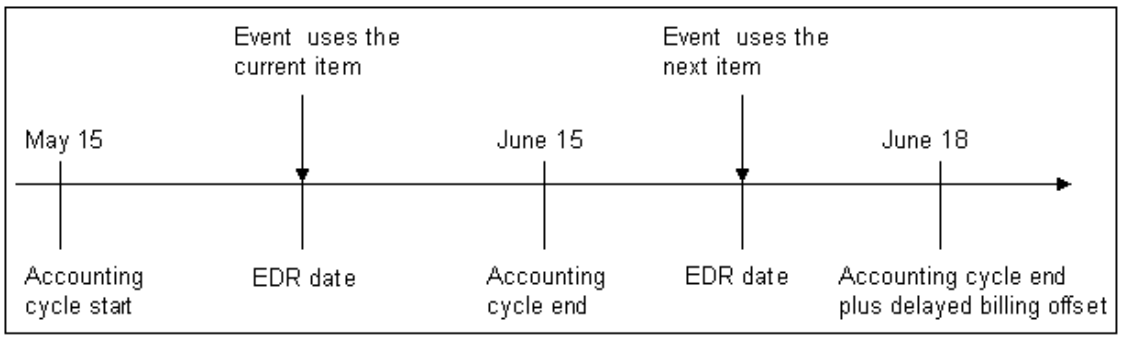
> **Note**
>
> You can set up an accounting cycle delay period to manage which items delayed events are assigned to. This is useful when your billing cycle spans multiple accounting cycles. See "About Accounting Cycle Delay Periods".

To assign the event to an item:

- If the EDR date falls before the next accounting date, the event is assigned to the current item.
- If the EDR date falls after the next accounting date, the event is assigned to the next item. This can happen because the event might occur after the close of the accounting cycle but before the delayed billing offset date.

Figure 7-4 shows how events are assigned.

Figure 7-4 Event Assignment

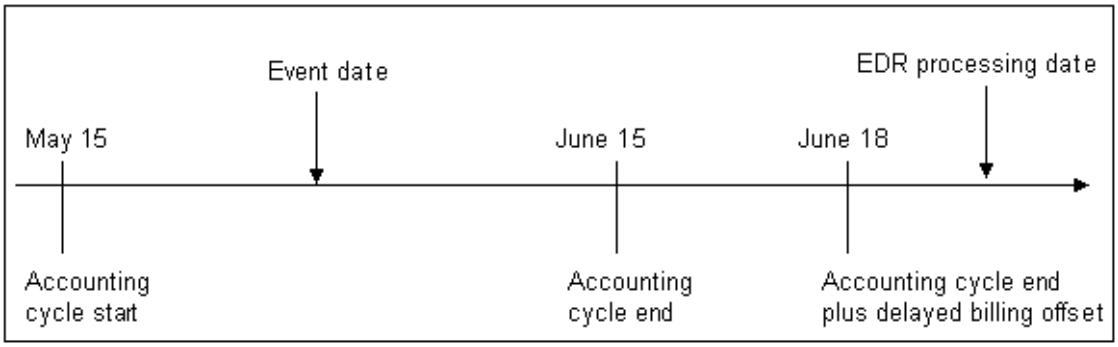


> **Note**
>
> The customer billing date is not relevant when choosing which item to use for the event. There might be multiple accounting cycles in one billing cycle. New items are created for each accounting cycle.

Copyright © 2025, 2026, Oracle and/or its affiliates.

If Pipeline Manager needs to assign an event to the current item, but billing for that item has already occurred, Pipeline Manager includes the event and assigns it to the current item. For example, if the event is rated on May 20 and loaded after the account cycle ends, it is still included in the current item as shown in Figure 7-5.

Figure 7-5 Current Event Assignment After Billing



About Accounting Cycle Delay Periods

In the batch pipeline, events that occur in one cycle can sometimes be rated and loaded into the BRM database after that cycle is completed. To assign delayed events to items of the billing cycle in which they occurred, you configure delayed billing. To assign delayed events to items of the accounting cycle in which they occurred, you configure an accounting cycle delay period.

When a billing cycle spans multiple accounting cycles, the items for those accounting cycles are not closed until billing is run. If you run a general ledger (G/L) report at the end of an accounting cycle for which billing has not yet been run, the status of the revenue in the G/L can change if additional events are rated and loaded for that cycle before the accounts are billed.

If you require that G/L data not change after the G/L report is run, you can configure an accounting cycle delay period after which events are no longer assigned to items of that accounting cycle, even if those items are not closed. You then run G/L reports after the accounting cycle delay period has ended. This ensures that the revenue reported in the G/L is accurately represented and that the state of the revenue (earned, unearned, billed, and unbilled) does not change after the G/L report is run.

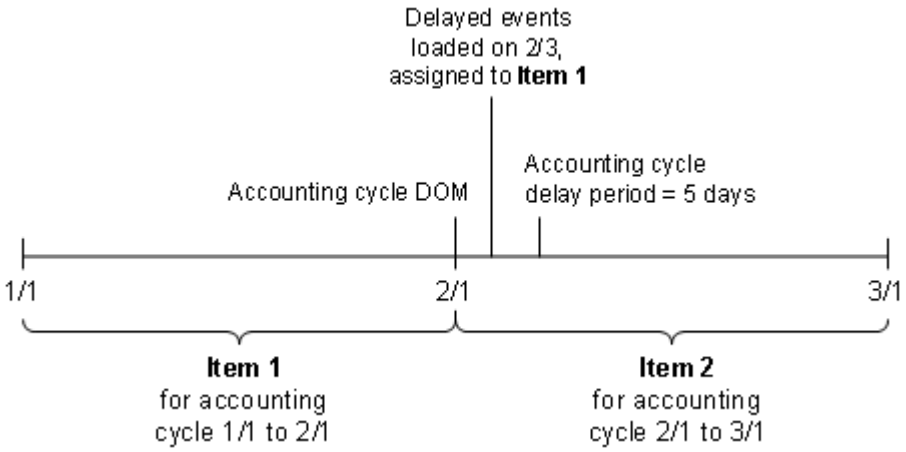
> **Note**
>
> To use an accounting cycle delay period, you must also configure delayed billing.

When you configure an accounting cycle delay period, BRM assigns delayed events to items based on when the accounting cycle delay period ends. BRM assigns events to items when RE Loader loads the events into the database:

- When delayed events (events that occurred in the previous cycle) are loaded after the accounting day of month (DOM), but before the end of the accounting cycle delay period, those events are posted to the item for which the DOM has just passed as shown in Figure 7-6:

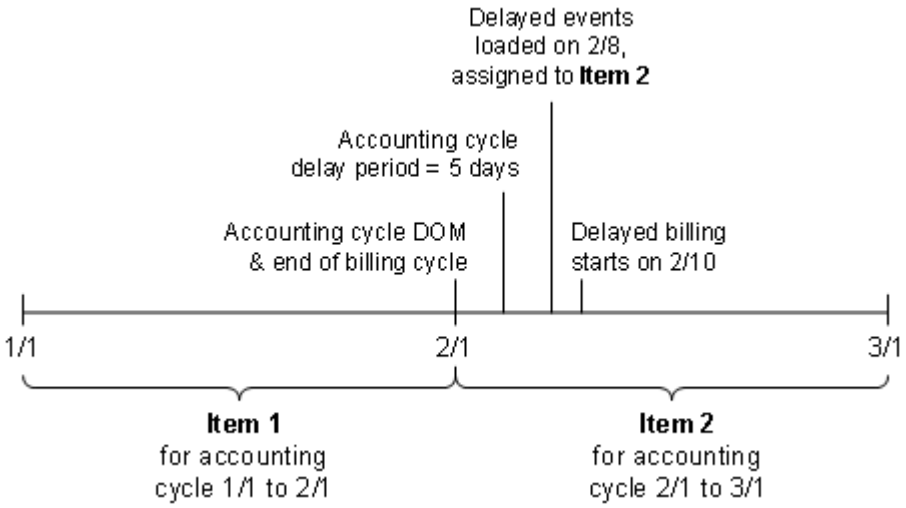
Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 7-6 Delayed Events Arriving During Cycle Delay Period



- When the billing cycle has ended and delayed events are loaded after the end of the accounting cycle delay period, but before delayed billing is run, those events are posted to the item for the next (current) accounting cycle, even though the previous cycle has not been billed and its items are still pending. This is shown in Figure 7-7:

Figure 7-7 Delayed Events Arriving After Cycle Delay Period



- After the account is billed, items for the billed cycle are closed so delayed events are posted to the item for the following (the current) cycle.

> **Note**
>
> If the accounting cycle delay period is longer than the delayed billing period, the accounting cycle delay period is ignored after billing is run. After billing is run, if any remaining events that occurred in the previous cycle are rated and loaded in the current cycle, they are assigned to the current cycle's item.

Copyright © 2025, 2026, Oracle and/or its affiliates.

You specify the accounting cycle delay period by modifying a business parameter configuration (**/config/business_params** object). RE Loader checks the accounting cycle delay period in the **/config/business_params** object before loading events. See "Configuring an Accounting Cycle Delay Period".

Configuring an Accounting Cycle Delay Period

By default, an accounting cycle delay period is disabled. You can enable this feature by modifying a field in the **billing** instance of the **/config/business_params** object.

You modify the **/config/business_params** object by using the **pin_bus_params** utility.

To configure an accounting cycle delay period:

1. Use the following command to create an editable XML file from the **billing** instance of the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsBilling bus_params_billing.xml
   ```

   This command creates the XML file named **bus_params_billing.xml.out** in your working directory. The file contains the current billing configuration values in the **/config/business_params** object in the BRM database. If you don't want this file in your working directory, specify the path as part of the file name.

2. Open **bus_params_billing.xml.out** file and search for the following line:

   ```
   <AcctCycleDelayPeriod>-1</AcctCycleDelayPeriod>
   ```

3. Change **-1** to the number of days in the accounting cycle delay period. The number of days must be a positive value. (A value of **-1** indicates that there is no accounting cycle delay period.)

   For example, if the accounting cycle delay period is 3 days and the accounting cycle ends at midnight on March 31, the delay period ends at midnight on April 3.

   > **Note**
   >
   > - BRM uses the XML in this file to overwrite the existing **billing** instance of the **/config/business_params** object. If you delete or modify any other parameters in the file, these changes affect the associated aspects of the BRM billing configuration.
   > - Do not set the accounting cycle delay period to be longer than the delayed billing period.

4. Save and close the **bus_params_billing.xml.out** file and rename the file to **bus_params_billing.xml**.

5. Use the following command to load this change into the **/config/business_params** object:

   ```
   pin_bus_params bus_params_billing.xml
   ```

   You should run this command from the *BRM_home*/**sys/data/config** directory, which includes support files used by the utility. *BRM_home* is the directory where you installed BRM components.

6. Read the object by using the **testnap** utility or Object Browser to verify that all fields are correct.

You do not need to restart the Connection Manager (CM) to update this entry.

Copyright © 2025, 2026, Oracle and/or its affiliates.

About G/L IDs

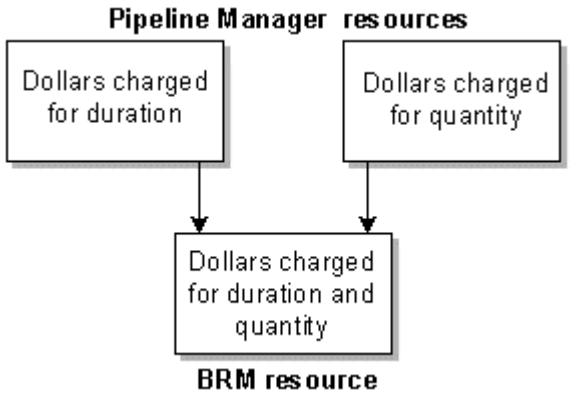
Pipeline Manager uses G/L IDs defined in the Pipeline Manager database. However, those G/L IDs must match the G/L IDs in the BRM database. You should define G/L IDs in the BRM database first and then in Pipeline Manager.

About Mapping Balance Elements between the Pipeline Manager Database and the BRM Database

Balance Element IDs must match between the Pipeline Manager database and the BRM database. In addition, you can configure how Pipeline Manager balance elements map to BRM balance elements. For example, Pipeline Manager can separate balance element amounts in a variety of ways, but you might want to combine balance element amounts when defining BRM balance impacts.

Figure 7-8 shows how the charges for a GPRS event can be mapped between the Pipeline Manager database and the BRM database:

Figure 7-8 GPRS Charge Mapping Between BRM and the Pipeline Manager



How Pipeline Manager Gets Historical Data

Because there is a gap of time between when a call occurs and when it is rated, information about the customer can change during that time. For example, a customer might change the phone number before a call is rated. Pipeline Manager needs to look up account data based on the old number.

To retrieve historical information, Pipeline Manager gets data from audited objects. By default, auditing in BRM is turned off for most objects. After you install BRM, run the **object_auditing.pl** script to turn on auditing for the objects and fields that Pipeline Manager needs data about.

In addition, Pipeline Manager pricing configuration data includes validity dates that can be used to apply the correct rating to delayed events.

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Loading Pipeline-Rated Event Data

Pipeline Manager sends the results of rating to output files. BRM loads the data in these files into the BRM database.

You configure BRM Batch Controller to start a batch handler when rated event files are ready for loading.

You configure a batch handler to load the rated-event data into the BRM database. The batch handler runs the utilities that load the data.

BRM provides the following sample batch handlers that you can use to load pipeline-rated events:

- **SampleRelHandler**: This batch handler starts the Rated Event (RE) Loader utility (pin_rel). The utility loads data for events that are rated in batches.
- **OODHandler**: This batch handler starts the rerate-request loader utility (pin_load_rerate_jobs). The utility creates rerate jobs for events that were rated out of order.
- **SampleHandler**: This batch handler starts the Universal Event (UE) Loader utility (uel). The utility loads batches of events into the BRM database for rating by the rating opcodes. UE Loader is also used by Pipeline Manager to load various types of data files that it generates during rating; for example, validity period data for products/charge offers, discounts/discount offers, and balance elements that start on first usage, roaming settlement data, and revenue assurance data.
- **ConfigurableValidityHandler**: This batch handler starts the **pin_rel** and **pin_load_rerate_jobs** utilities, and the instance of **uel** that loads first-usage validity data, thereby eliminating the need to configure separate instances of the handlers for these utilities. For more information, see "About Using a Single Batch Handler to Run Multiple Loading Utilities".

About Using a Single Batch Handler to Run Multiple Loading Utilities

If you use Pipeline Manager to rate products/charge offers, discounts/discount offers, or granted balance elements that start on first usage, you configure the ConfigurableValidityHandler sample batch handler to run the utilities for RE Loader (**pin_rel**), UE Loader (**uel**), and out-of-order rerating (**pin_load_rerate_jobs**).

> **Note**
>
> If you do not wish to use a single batch handler to run these utilities, you can configure the individual sample batch handlers provided for RE Loader, UE Loader, and out-of-order rerating.

ConfigurableValidityHandler runs **pin_rel**, **uel**, and **pin_load_rerate_jobs** instances sequentially, waiting for one process to complete before starting the next. In a single sequence, ConfigurableValidityHandler processes files that were produced for the same CDR file or for the same pipeline transaction (if the pipeline is configured to produce an output file per transaction instead of per CDR file).

When pipeline rating outputs a file of rated events, the ConfigurableValidityHandler batch handler performs the following tasks:

Copyright © 2025, 2026, Oracle and/or its affiliates.

1. Runs **pin_rel** to load the results of pipeline rating into the BRM database.

   For more information about RE Loader, see "About Loading Rated Events into the BRM Database" in *BRM Loading Rated Events*.

2. Checks for first-usage validity files for products/charge offers and discounts/discount offers and, if one exists for the same CDR or transaction, runs **uel** to load the product/charge offer and discount/discount offer validity data.

   For more information about using UE Loader to load validity data, see "About Updating Validity Period Information in the BRM Database" in *BRM Setting Up Pipeline Rating and Discounting*.

3. Checks for first-usage files for balance elements and, if one exists for the same CDR or transaction, runs **uel** again to load the balance element validity data.

4. Checks for rerate-request files for events rated out of order and, if one or more exist for the same CDR or transaction, runs **pin_load_rerate_jobs** for each file.

ConfigurableValidityHandler handles errors in the following ways:

- If **pin_rel** does not successfully load the pipeline batch-rated events, the failure is logged in the handler log file (**configurable_validity_handler.log**) and **uel** and **pin_load_rerate_jobs** are not run.
- If **uel** or **pin_load_rerate_jobs** fails, the entire process is recorded as failed in the handler log file.

If all processes are successful, the **configurable_validity_handler.log** file is deleted. If one or more processes are not successful, the **configurable_validity_handler.log** file is not deleted.

Along with processing information and status, the log files include the name of the input file that was loaded. If **pin_rel** loads a rated-event file and there were no associated first-usage validity files or rerate-request files, this is noted in the log file.

About Pipeline Rating and BRM Billing

When Pipeline Manager receives an EDR for the next billing cycle for an account that hasn't yet been billed, the EDR is suspended (not rated). Only when the account's billing process is complete can the new EDRs be rated.

The number of accounts being billed affects the time it takes to complete the billing process. The longer the processing time, the greater the chance EDRs might need suspending or rerating. If you process many EDRs and need accounts to be billed quickly so that their new usage can be rated, you can set up Pipeline Manager to trigger billing. When Pipeline Manager triggers billing for an account, it is billed in a separate billing process.

Function Module Dependencies

Table 7-1 provides guidelines for how to configure the order of function modules in a pipeline. The modules are listed in a typical order, but your configuration might vary. Some modules require that other modules be run first, whereas some modules can be located anywhere in the pipeline.

For more information, see the reference documentation for the module.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 7-1 Functional Modules and Processing Dependencies

Function module	Processing dependencies
FCT_PreSuspense	Must be the first preprocessing module in a pipeline.
ISC_SetAndValidateBatchInfo	This is the first iScript that must be used so that the batch ID gets inserted before any further processing of the mediation batches.
FCT_DuplicateCheck	Should run early in a pipeline to discard duplicate EDRs.
FCT_CallAssembling	Should run early in a pipeline to assemble EDRs. Must run before FCT_Discard.
FCT_ServiceCodeMap	Some modules require an internal service code, so this module should run near the front of a pipeline.
ISC_CiberInputValidation	Because erroneous CIBER records can be discarded, this module must run before the FCT_Discard module.
FCT_Discard	Because you can discard or split EDRs based on service codes, this module should run after the FCT_ServiceCodeMap module. Should be early in the function pool, but must be run after FCT_CallAssembling.
iRuleValidation	Run this iRule before ISC_TapSplitting.
ISC_TapSplitting	Must run after the following modules: • FCT_DuplicateCheck • FCT_Discard
FCT_AccountRouter	For general use, this module must run after the FCT_ServiceCodeMap module and before the rating modules. For use with standard recycling or Suspense Manager using multiple database schemas, this module must run before OUT_GenericStream in a pre-recycling pipeline. This module sends output to a separate pipeline for each BRM database schema.
FCT_EnhancedSplitting	Because you can split EDRs based on service codes, this module should run after the FCT_ServiceCodeMap and FCT_UsageClassMap modules.
FCT_CliMapping	Must run before the rating modules.
FCT_UoM_Map	Must run after the FCT_ServiceCodeMap module and before the rating modules.
FCT_UsageClassMap	Must run before the zoning and rating modules.
FCT_NumberPortability	Must run before the zoning and rating modules.
ISC_MapNetworkOperatorInfo	Must run after the FCT_NumberPortability module and the ISC_PopulateOpcodeAndUtilBlock_Diameter iScript.
FCT_NOSP	Must run before segment rating is performed.
FCT_Account	Must run before the zoning and rating modules.
ISC_ProfileAnalyzer	Must run after FCT_Account and before any rating modules.
ISC_ProfileLabel	Must run after FCT_Account and before any rating modules.
IRL_UsageType	Must run after FCT_Account and before FCT_USC_Map.
FCT_CiberOcc	Must run after the FCT_DuplicateCheck module and before the FCT_CarrierIcRating module.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 7-1 (Cont.) Functional Modules and Processing Dependencies

Function module	Processing dependencies
FCT_ItemAssign	Must run after the FCT_Account, rating, and discounting modules and before the FCT_BillingRecord module.
FCT_CustomerRating	Must run after FCT_Account.
FCT_Filter_Set	Must run after FCT_Account.
IRL_PromotionalSavingPerEDR	Must run before IRL_LeastCost and FCT_CustomerRating.
IRL_LeastCostPerEDR	Must run before FCT_CustomerRating and *after* "FCT_Filter_Set".
FCT_CustomerRating	Must run after FCT_Account.
ISC_LeastCost	Must run after FCT_CustomerRating.
FCT_CiberOcc	Must run after the FCT_DuplicateCheck module and before the FCT_CarrierIcRating module.
FCT_CarrierIcRating	Must run after FCT_Account.
FCT_DroppedCall	Must run after FCT_Account.
FCT_APN_Map	Can run before or after the zoning modules (FCT_Zone and FCT_PreRating).
FCT_PreRating	Must run before the FCT_MainRating module.
FCT_SegZoneNoCust	Must run after the prerating module
FCT_MainZoning	Must run after the FCT_SegZoneNoCust module.
FCT_Zone	Must run after FCT_Account.
FCT_USC_Map	Must run after the following: • FCT_UsageClassMap • ISC_UsageType • FCT_PreRating
FCT_TriggerBill	Must run before the FCT_MainRating module.
FCT_MainRating	Must run after at least one of the following modules: • FCT_GlobalRating • FCT_CustomerRating • FCT_CarrierIcRating
FCT_RSC_Map	Must run after the FCT_MainRating module to adjust the rate.
FCT_Dayrate	Must run after the FCT_MainRating module to adjust the rate.
FCT_RateAdjust	Must run after the FCT_MainRating module to adjust the rate.
FCT_Rounding	Must run after the FCT_RateAdjust module if you want rating results to be rounded and after FCT_Discount module if you want discount results to be rounded. FCT_Rounding must come after each module for which rounding should occur. For batch rating, it must come before the FCT_ApplyBalance module.
FCT_ExchangeRate	Must run after the FCT_MainRating module.
ISC_FirstProductRealtime	Must run this iScript in the real-time rerating pipeline before the FCT_DiscountAnalysis module.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 7-1 (Cont.) Functional Modules and Processing Dependencies

Function module	Processing dependencies
FCT_DiscountAnalysis	For pipeline rating, must run after the FCT_Account module and before the FCT_Discount module. For real-time rating, must run before the FCT_Discount module.
FCT_Discount	Must run after the FCT_MainRating module. For batch rating, this module should be in the same function pool as the FCT_ApplyBalance module and run before that module.
FCT_FirstUsageNotify	Must run this module before the FCT_ApplyBalance and FCT_Reject modules.
FCT_ApplyBalance	Must run after the FCT_Rounding module. This module should be in the same function pool as the FCT_Discount module and run after that module.
ISC_TaxCalc	Must Run this module before the FCT_BillingRecord module, but after the FCT_MainRating module.
FCT_BillingRecord	Must run after the FCT_MainRating and FCT_Discount modules.
FCT_EventOrder	Must run *after* the FCT_MainRating and FCT_Discount modules and *before* the FCT_Reject module.
ISC_CiberOutputMapping	Must run after the FCT_MainRating module and ISC_PostRating iScript.
IRL_EventTypeSplitting	Must run after FCT_ServiceCodeMap and before the FCT_Reject module. This is typically the last module before the FCT_Reject module.
FCT_Reject	Runs after the rating and discount modules.
ISC_SetOutputStream	Must run after FCT_Reject.
ISC_PostRating	Must run: • After rating modules FCT_CustomerRating, FCT_PreRating, and FCT_MainRating or • After the FCT_ExchangeRate module
ISC_SetEDRStatus	Must run before FCT_AggreGate.
ISC_SetRevenueFigures	Must run after rating and discounting and before FCT_AggreGate.
ISC_SetRevenueStream	Must run before FCT_AggreGate and after post rating (after the EDRs are rated).
ISC_SetEDRStatus	Must be run before the FCT_AggreGate scenario that collects audit data grouped on the EDRStatus field.
FCT_AggreGate	Runs after rating modules. This module usually runs in its own pipeline.
FCT_CancelTimer	Depends on the FCT_Timer in the Dispatcher pipeline for the TimerID and the timeout flag values.
FCT_Recycle	For pipeline-only recycling, must be the last module in the pipeline.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 7-1 (Cont.) Functional Modules and Processing Dependencies

Function module	Processing dependencies
FCT_Suspense	For standard recycling and Suspense Manager, must be the last module in a pipeline.

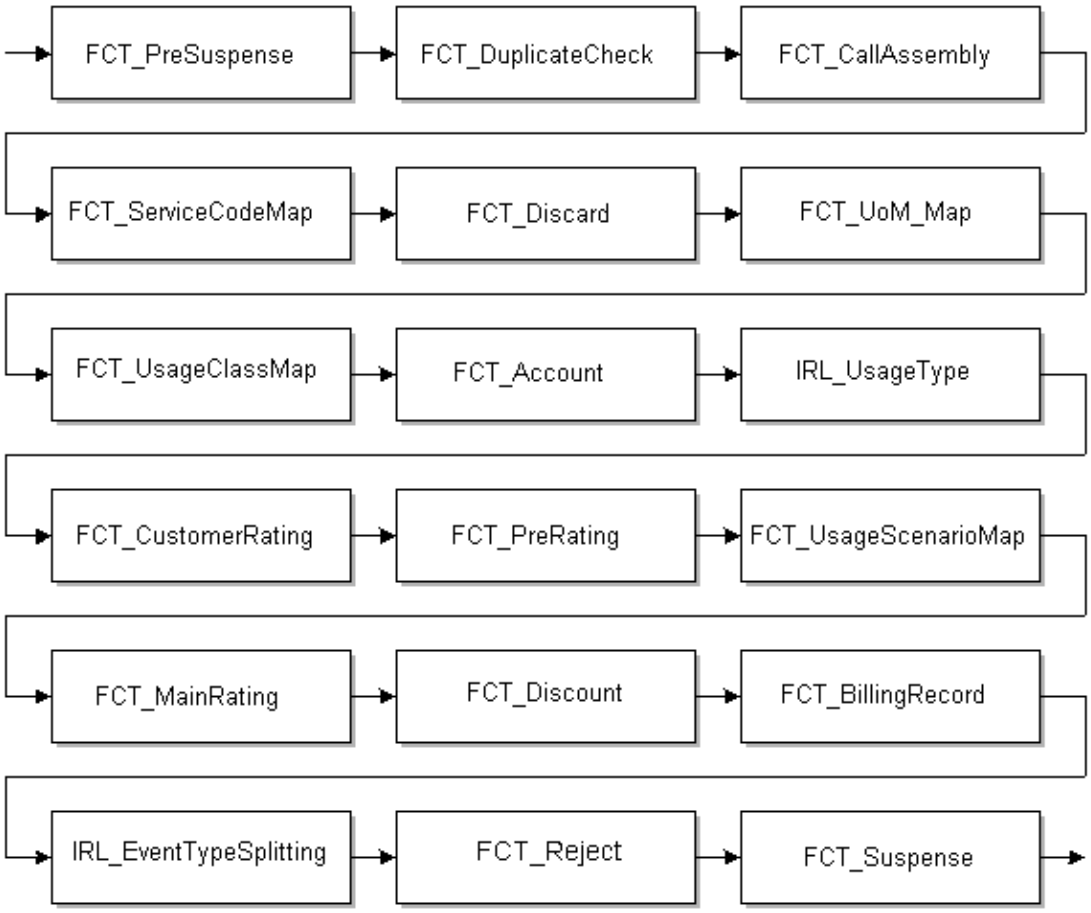
The following modules in *BRM Pipeline Manager Reference* do not have any placement dependencies and can be run anywhere in a pipeline or in a separate pipeline, depending on the data that is being processed:

- FCT_IRules
- FCT_IScript
- ISC_AddCBD
- IRL_PipelineSplitting
- FCT_Timer
- FCT_PreRecycle
- FCT_Opcode
- ISC_ApplyTax
- ISC_BACKOUTTypeSplitting
- ISC_CiberRejectReason
- ISC_ConsolidatedCP
- ISC_DupRAPRecords
- ISC_EDRToTAPOUTMap
- ISC_Migration
- ISC_MiscOutcollect
- ISC_ObjectCacheTypeOutputSplitter
- ISC_OverrideSuspenseParams
- ISC_RemoveASSCBD
- ISC_RollbackSettlement
- ISC_UsageClassSetting
- FCT_SocialNo
- FCT_PrefixDesc

Figure 7-9 shows the order of the most common function modules:

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 7-9 Order of Common Function Modules



Data Module Dependencies

This section provides guidelines for how to configure the order of data modules in the registry file. Some modules require that other modules be run first, whereas some modules can be located anywhere in the pipeline.

- "DAT_PortalConfig" in *BRM Pipeline Manager Reference*. Due to the dependency of other data modules on DAT_PortalConfig, the DAT_PortalConfig registry entries must appear before all other data module entries in the registry file.

Copyright © 2025, 2026, Oracle and/or its affiliates.

8

Policy-Driven Charging

This chapter describes how to manage policy-driven charging of services by using Oracle Communications Billing and Revenue Management (BRM) real-time and offline rating.

Topics in this document:

- About Policy-Driven Charging in BRM
- About Setting Up Policy-Driven Charging
- How Policy-Driven Charging Works
- About Notifications Related to Offer Profiles Sent by BRM
- Configuring Your BRM Environment to Support Policy-Driven Charging of Services
- About Setting Up and Managing Offer Profiles
- About Configuring Resources for Tracking by BRM
- Configuring Event Notification for Policy-Driven Charging
- Enabling Enterprise Applications to Handle Policy-Driven Charging Events
- Updating Reservation Preferences Configuration for Policy-Driven Charging
- Updating the Connection Manager for Policy-Driven Charging
- Customizing Information Received from BRM
- Managing Offer Profiles with Opcodes
- Customizing Subscriber Preferences Data for Policy-Driven Charging
- Customizing Business Events Associated with Policy-Driven Charging
- Sample Threshold Breach Notification

About Policy-Driven Charging in BRM

Policy-driven charging enables you to track a subscriber's service usage and, based on that usage, change the customer's quality of service (QoS) in real-time rating and in offline rating for offline events such as billing and adjustments.

For example, a subscriber purchases a plan for a certain QoS to download video content. The subscriber makes his choice from one of many plans that you have configured with gradations in the QoS based on usage amounts in megabytes, such as 100-150, 150-200, and 200-250 megabytes. When the subscriber starts downloading video content from the network, you can track the quantity of megabytes that the subscriber downloads during that session. When you find that the downloaded quantity crosses the upper threshold set for the selected QoS (for example 150 megabytes), you can use BRM's policy-driven charging to make a seamless change in the policy set for the subscriber's (video downloading) session on the network and allow a change to the QoS from the current to the next level.

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Setting Up Policy-Driven Charging

To set up policy-driven service charging:

- Create offer profiles to define ranges in the QoS for the services you offer, and include the offer profiles in your price lists.

  For example, set up high, medium, and low QoS for downloading data from the network, using a noncurrency resource called *Megabytes Used*. Create an offer profile with a unique name (for example, *Platinum*). In the offer profile, specify the resource ID **100009** for the noncurrency resource and set the ranges to 100-150, 150-200, and 200-250 megabytes.

- Link the offer profile to the product or discount with which it is associated so that BRM can track the usage correctly.

  For example, connect the offer profile called *Platinum* to a product or discount by using *Platinum* as the provisioning tag for the product or discount. Use the resource ID **100009** for the noncurrency resource *Megabytes Used* configured in the offer profile as the noncurrency resource in the rate plan for the product or discount.

- Set up provisioning rules that state how the network is to provide and manage the services you offer.

  For example, when a subscriber's usage of a video downloading service reaches or crosses a threshold set in the offer profile (for example,150 megabytes of data download), a different rule can be enforced for the network session, based on the subscriber's profile, his balances, his usage of the resource, and the traffic on the network.

  You configure these provisioning rules in the application you use to provide network policy control. Policy controllers (such as Oracle Communications Policy Controller) manage the QoS, optimize high bandwidth traffic, and enforce usage quota levels for subscribers using the network.

  The policy controller installs the provisioning policy on the network element responsible for managing the enforcement of network policies on traffic passing through the network.

> **Note**
>
> Setting up provisioning rules to use in policy-driven charging for network capacity and services is beyond the scope of this document. For more information, see the appropriate references.

In addition to configuring offer profiles for provisioning rules in the policy controller, you must configure the following:

- Subscriber preferences for communications, such as whether the subscriber wants to be notified, in what language, and so on.
- Other factors (such as rates) that you use to give better services to more valuable subscribers when network capacity is limited during peak time or in an area.
- A network connectivity application to start, monitor, and end the online charging services for sessions in the network.

  Network connectivity applications (such as Oracle Communications Online Mediation Controller) communicate with BRM to manage charging sessions.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> Setting up the network connectivity application for policy-driven charging is beyond the scope of this document. For more information, see the appropriate references.

- Event notification to trigger policy-driven charging operations for real-time charging.

How Policy-Driven Charging Works

BRM uses the data you configure in offer profiles to provide you with offer profile threshold breach notifications for both in-session and out-of-session rating events.

Suppose you configure an offer profile called *Platinum* with specific usage ranges for downloading video content. The noncurrency resource such as *Megabytes (MB) Used* and its usage ranges such as *100-150* megabytes, at *150-200* megabytes, over *200-250* megabytes are defined within a policy label (called *Fair Usage*).

When your subscriber initiates the downloading of a video,

- The policy controller communicates with BRM to set up a provisioning policy for the session.

  A provisioning rule is established on the network for the session for an initial quantity, for example, 35 megabytes. The range for the current QoS for the subscriber is 100-150 megabytes.

- The download begins with the network connectivity application managing the online-charging session.

Throughout that session, whenever the subscriber requests more quota, BRM calculates whether it can allocate the requested amount or if it must readjust the quota.

For example, BRM receives a request with the information that the subscriber has used 35 megabytes and needs 75 megabytes more. BRM does the following:

- Calculates how much the subscriber has used.

  BRM uses the balance information and consumed reserved amount (across parallel sessions, iPhone, video, and computer) for that subscriber. For example, BRM calculates that the subscriber has used a total of 150 megabytes. With 100-150 megabytes as the range for the current QoS for the subscriber, his usage is at the threshold for the policy.

- Reduces the allocation, if necessary.

  In the example case, the subscriber has used a total of 150 megabytes. The upper thresholds of the offer profiles are set at 150, 200, and 250 megabytes. The next threshold is 200 megabytes. The available quota for that QoS is 50 megabytes only (200-150). The subscriber has requested 75 megabytes, an amount which is greater than the available quota of 50 megabytes.

  In this scenario, BRM readjusts the quota allocation to 50 megabytes.

When the used amount for the resource crosses a set threshold (150, 200, 250, and so on), the provisioning rule (which was tied to the previous policy level) can no longer be applied for the session.

BRM and the policy controller ensure that a more appropriate rule is established for the session on the network in the following way:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- BRM sends an offer profile threshold breach notification to the policy controller with information on the threshold breach (150 in our example case) and information on the subscriber's preferences for receiving communications.
- The policy controller does the following:
  - Communicates with BRM to retrieve the latest information on the subscriber and his usage of the service.
  - Arrives at the new rule which enables the increase in bandwidth.
  - Establishes that provisioning rule for the session.

About Notifications Related to Offer Profiles Sent by BRM

When you set up policy-driven charging for your services, BRM sets up notification events for the following:

- When you create, modify, or delete the following:
  - Offer profiles
  - Products associated with offer profiles
  - Discounts associated with offer profiles
- Offer profile threshold breaches

  Offer profile threshold breaches can occur with your subscriber's usage of a resource in a prepaid session. Or, they can occur when BRM processes offline events, such as billing and adjustments.

BRM places these offer profile threshold breach notifications in a central Oracle Advanced Queuing (AQ) database queue for use by customer relationship management (CRM) applications. You can use these applications to retrieve data directly from the Oracle AQ database queue and use the information in these offer profile threshold breach notifications. See *BRM Developer's Guide* for more information on the Oracle AQ database queue.

BRM provides the following information in offer profile threshold breach notifications associated with policy-driven charging:

- Service identifier
- List of aliases that also identify this user/service (ALIAS_LIST)
- Name of the resource
- Resource ID
- Status label from the offer profile

See "Sample Threshold Breach Notification".

Configuring Your BRM Environment to Support Policy-Driven Charging of Services

Configure your BRM environment to support policy-driven charging of services by completing the following if you are using Pricing Center and Pipeline Manager:

1. Set up offer profiles in your price lists. See "About Setting Up and Managing Offer Profiles".
2. Configure BRM to track a subscriber's usage of the resources you offer. Set up BRM to create notifications tailored to the subscriber's preferences on how they want to be notified

Copyright © 2025, 2026, Oracle and/or its affiliates.

when their usage exceeds the threshold set in their offer profiles. See "About Configuring Resources for Tracking by BRM ".

3. Merge the event notification entries specific to policy-driven charging with the BRM event notification list. See "Configuring Event Notification for Policy-Driven Charging".
4. Enable your enterprise applications to integrate with policy-driven charging in BRM. See "Enabling Enterprise Applications to Handle Policy-Driven Charging Events".
5. Update the BRM database with reservation preferences configuration for policy-driven charging. See "Updating Reservation Preferences Configuration for Policy-Driven Charging".
6. Ensure that the Connection Manager can forward the notifications associated with policy-driven charging. See "Updating the Connection Manager for Policy-Driven Charging".

About Setting Up and Managing Offer Profiles

You can create offer profiles when you create a price list or add your offer profile data to an existing price list. See "Setting Up Offer Profiles" for information on setting up the data for offer profiles.

About Using New Resource IDs in Your Offer Profiles

When you start (or restart) your BRM system, BRM caches the set of unique resource IDs that you have configured in your offer profiles. BRM processes a subscriber's usage only if the resource ID is present in its cache.

> **Note**
>
> When you introduce a resource ID that has not been used in BRM, you must update the entries in this cache. To do so, stop and start the Connection Manager (CM).

Configuring Offer Profile Data

BRM provides sample offer profile data in the default **pin_offer_profile_policy_labels.xml** file.

This sample file uses the format detailed in the *BRM_home***/PricingCenter/price_list.xsd** file. *BRM_home* is the directory in which you installed the BRM software. Maintain your offer profiles in the format used in the sample file. For more information on using the XML interface, see "Using the XML Pricing Interface to Create a Price List ".

> **Note**
>
> - Offer profile names must be unique.
> - If you introduce a new resource ID in an offer profile you add to or modify in the database, you must restart the CM after you load the offer profiles.

To configure your offer profile data:

1. Go to the *BRM_home***/setup/scripts** directory.

Copyright © 2025, 2026, Oracle and/or its affiliates.

2. Open the **pin_offer_profile_policy_labels.xml** file in an XML editor or text editor.

> **Tip**
>
> Save a copy of the default file before you make any changes to it.

3. Edit this file using the parameters as shown in the file. Table 8-1 describes the elements.

Table 8-1 Elements Used to Store Offer Profiles

Element	Description
label	Name of the policy label
offer_profile	Array to hold offer profiles
offer_profile_code	Key identifier associated with an offer profile
offer_profile_name	Name of an offer profile
policy_label	Array to hold policy labels
price_list	Price List object within which the offer profile array resides
resource_id	Id used for the resource associated with a policy label
resource_name	Name of the resource associated with a policy label
status_label	Status label name indicating the level of the tier
tier	The tier as an array with each entry made up of a status label, and the start and end of the range for that tier level.
tier_end_range	Start of the range for a tier level
tier_start_range	End of the range for a tier level
unit	Unit of the resource associated with a policy label. The values are: • None. (**0**) • Second (**2**) • Minute (**1**) • Hour (**3**) • Day (**4**) • Byte (**11**) • Kilobyte (**12**) • Megabyte (**13**) • Gigabyte (**14**) The default value is **0**.

4. Save the **pin_offer_profile_policy_labels.xml** file.

You are now ready to store the offer profiles in the BRM database.

Managing Offer Profile Data

BRM stores offer profiles in **/offer_profile** objects. Use the **loadpricelist** utility to load, retrieve, modify, and delete **/offer_profile** objects.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> Before you use this utility, verify that:
>
> - The XML offer profile file is complete and follows the guidelines.
> - The **Infranet.properties** file in your CLASSPATH has the correct login information for the BRM database to which you want to connect.
>
> For more information, see "Prerequisites for the XML Pricing Interface".

Storing Offer Profile Data

> **Note**
>
> If you introduce a new resource ID in an offer profile you add to the database, you must restart the CM after you load the offer profiles.
>
> To do so, stop and restart the CM.

Use the following command to load offer profiles into your BRM database:

```
loadpricelist -f -v -c pin_offer_profile_policy_labels.xml
```

where:

- **`-f`** runs the command without prompting for a confirmation.
- **`-v`** displays information about successful or failed processing as the utility runs
- **`-c`** reads the offer price information from **pin_offer_profile_policy_labels.xml** and commits it to the BRM database.

The offer profile information is now stored in **/offer_profile** objects in the database.

Loading Modified Offer Profiles in the Interactive Mode

You can commit the modified offer profile data by using the **loadpricelist** utility when you are working in the interactive mode.

To load offer profile data only:

```
write PriceListFile offer_profile PriceObject
```

where:

- *PriceObject* is the specified pricing object in the database.
- *PriceListFile* is the entire price list in the database.

For more information on the utility, see "loadpricelist".

Retrieving Offer Profiles

To retrieve offer profile data, use the **loadpricelist** utility in the following ways:

- Non-Interactive mode:

Copyright © 2025, 2026, Oracle and/or its affiliates.

```
loadpricelist -r PriceListFile
```

- Interactive mode:

```
read PriceListFile
```

The complete price list is returned in the *PriceListFile* XML file. Use your custom code to retrieve the **offer_price** objects contained in that price list.

Modifying Offer Profiles

> **Note**
>
> If you introduce a new resource ID in an offer profile you modify, you must restart the CM after you load the offer profiles.

To modify the offer profile data in the BRM database:

1. Retrieve the offer profile data and use your custom code to retrieve the **offer_price** objects. See "Retrieving Offer Profiles".
2. Modify the **offer_price** objects as necessary.
3. Configure the offer profile data in XML format. See "Configuring Offer Profile Data".
4. Commit the modified offer profile data into the database. See "Storing Offer Profile Data".
5. If you added a new resource ID to any offer profile, stop and restart the CM.

Deleting Offer Profiles

Delete offer profiles using the **loadpricelist** utility in the following ways:

- Non-Interactive mode:

```
loadpricelist -p
```

This command deletes all the pricing objects from the database.

- Interactive mode:

```
delete offer_profile PriceObject
```

> **Note**
>
> If you use the **delete** command without any parameters, all in-memory pricing information is deleted from the database.

About Configuring Resources for Tracking by BRM

Configure the resources that BRM should track by setting up provisioning tags for your services as shown in the *BRM_home***/sys/data/config pin_offer_profile_provisioning_tags_policy_attributes.xml** file. For each provisioning tag, configure the types of resources valid for that tag and provide information on the specific opcode which must be called to process changes to the product or discount associated with that provisioning tag.

Copyright © 2025, 2026, Oracle and/or its affiliates.

The **pin_offer_profile_provisioning_tags_policy_attributes.xml** file contains the configuration for the default provisioning tag that BRM provides. See Example 10-1. The first section of this file lists the valid resources for the default provisioning tag. The next section contains information on the opcodes to be called when the product or discount containing the provisioning tag is purchased or canceled, the parameters that specify the fields to be added to the opcode's input flist, and the value for each field.

See "Configuring Provisioning Tags for Policy-Driven Charging" for more information.

Configuring Event Notification for Policy-Driven Charging

To configure event notification for policy-driven charging, you must merge the policy-driven charging event notification list with the general BRM event notification list.

Each event in an event notification list is mapped to the opcode or opcodes that are run when the event occurs. The event notification list for policy-driven charging contains of entries for offer profiles and for the products and discounts associated with them.

Example 8-1 shows the event notification list for offer profiles:

Example 8-1 Event Notifications Related to Offer Profiles

```
1301    0      /event/notification/offer_profile/create
1301    0      /event/notification/offer_profile/update
1301    0      /event/notification/offer_profile/delete
1301    0      /event/notification/offer_profile/ThresholdBreach
```

BRM uses the Enterprise Application Integration (EAI) framework publishing opcode, PCM_OP_PUBLISH_GEN_PAYLOAD (number 1301), to set up event notifications for offer profiles. This opcode is internal to BRM.

Example 8-2 shows the event notification list for products and discounts associated with offer profiles:

Example 8-2 Billing and Product Event Notification Entries

```
301     0       /event/billing/product/action/purchase
301     0       /event/billing/product/action/modify
301     0       /event/billing/product/action/cancel
301     0       /event/billing/product/action/modify/status
301     0       /event/billing/discount/action/purchase
301     0       /event/billing/discount/action/modify
301     0       /event/billing/discount/action/cancel
301     0       /event/billing/discount/action/modify/status
```

The **301** entry is the opcode number for the PCM_OP_ACT_POL_EVENT_NOTIFY policy opcode. See "About the PCM_OP_ACT_POL_EVENT_NOTIFY Policy Opcode".

For more information on event notification, see "Using Event Notification" in *BRM Developer's Guide*.

Merging the Policy-Driven Event Notification List with the BRM Event Notification List

To merge the policy-driven event notification list with the BRM event notification list:

1. Verify that the list entries for policy-driven charging are enabled. See "Verifying That Policy-Driven Charging Event Notification List Entries Are Enabled".

Copyright © 2025, 2026, Oracle and/or its affiliates.

2. Merge the list for policy-driven charging with the general BRM list. See "Merging Event Notification Lists".
3. Load the updated event notification data into the BRM database. See "Loading the Updated Event Notification File into the BRM Database".

Verifying That Policy-Driven Charging Event Notification List Entries Are Enabled

To verify that the policy-driven charging event notification list entries are enabled:

1. Open the *BRM_home*/**sys/data/config/pin_notify_ipc** file in a text editor.

> **Tip**
>
> Save a copy of the default file before you change it.

2. Verify the following:
   - The offer profile notifications are enabled as shown in Example 8-1.
   - The billing event notifications for product and discount objects are enabled as shown in Example 8-2.
3. Save and close the file.

For more information, see "Editing the Event Notification List" in *BRM Developer's Guide*.

Merging Event Notification Lists

Merge the event notification list in the **pin_notify_ipc** file with the event notification list in the **pin_notify** file. For more information, see "Merging Event Notification Lists" in *BRM Developer's Guide*.

Loading the Updated Event Notification File into the BRM Database

To enable event notification for policy-driven charging, run the **load_pin_notify** utility to load the merged event notification list into the BRM database. For more information, see "Loading the Event Notification List" in *BRM Developer's Guide*.

Enabling Enterprise Applications to Handle Policy-Driven Charging Events

Complete the following operations to enable your enterprise applications to handle policy-driven charging event notifications that BRM publishes:

1. Verifying That BRM Is Integrated with Your Enterprise Applications
2. Enabling Enterprise Applications to Process Policy-Driven Charging Events
3. Specifying the Payload Configuration File to Use

For more information, see "Integrating BRM with Enterprise Applications" in *BRM Developer's Guide*.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Verifying That BRM Is Integrated with Your Enterprise Applications

For your enterprise applications to use the offer profile event notifications published by BRM, ensure that BRM is integrated with other applications in your enterprise. To do so, ensure that EAI Manager is installed.

For more information, see "About Installing EAI Manager" in *BRM Developer's Guide*.

Enabling Enterprise Applications to Process Policy-Driven Charging Events

You enable your enterprise applications to receive the notifications that BRM publishes when it processes policy-driven charging events.

When a customer buys, modifies, cancels a product or discount that is associated with an offer profile, BRM publishes a business event (such as **ProductPurchase** when the customer buys a product). If an offer profile is created, modified, deleted or there is an threshold breach, BRM publishes a related business event (for example, **OfferProfileCreate**, when an offer profile is initially stored in the database).

Business events associated with policy-driven charging are defined in the *BRM_home***/sys/eai_js/payloadconfig_ipc.xml** payload configuration file. BRM provides default definitions of business events in the *BRM_home***/sys/eai_js/payloadconfig.xml** payload configuration file.

If you already use a payload configuration file (for example, **payloadconfig.xml**) in your BRM environment, set up one payload configuration file to contain the definitions of all your business events.

Setting Up One Payload Configuration File

Set up one payload configuration file by doing the following:

1. Merge the contents of **payloadconfig_ipc.xml** (the payload configuration file for policy-driven charging) with your current payload configuration file (such as, **payloadconfig.xml**).

   > **Tip**
   >
   > Save a copy of the default payload configuration files before merging them.

2. Save the merged file under a different name (for example, *your_payloadconfig.xml*).

Specifying the Payload Configuration File to Use

To specify the payload configuration file to use:

1. Open *BRM_home***/sys/eai_js/Infranet.properties** in a text editor.

   This is the payload generator external module **eai_js** properties file.

2. Point the **infranet.eai.configFile** entry to the appropriate payload configuration file.

   For example:

```
infranet.eai.configFile=/pinhome/pin201/opt/portal/15.0/sys/eai_js/your_payloadconfig.xml
```

   where *your_payloadconfig.xml* is the merged payload configuration file containing the business event entries associated with policy-driven charging.

Copyright © 2025, 2026, Oracle and/or its affiliates.

3. Save and close the file.
4. Restart **eai_js**.

Updating Reservation Preferences Configuration for Policy-Driven Charging

To update the reservation preferences configuration for services associated with policy-driven charging:

1. Add information on how the resource is to be rated (for example, by duration, by volume, by both duration and volume, or by occurrence). See "Updating Reservation Preferences Configuration Settings for Resources".

   > **Note**
   >
   > Policy-driven charging of services does not support prerated events.

2. Load the reservation preferences into the BRM database. See "Loading Reservation Preferences for Policy-Driven Charging".

Updating Reservation Preferences Configuration Settings for Resources

BRM stores the default entries for reservation preferences associated with policy-driven charging in the **pin_config_reservation_aaa_prefs_XXX** file. For example:

- **pin_config_reservation_aaa_prefs_gsm_telephony** (GSM telephony)
- **pin_config_reservation_aaa_prefs_gsm_data** (GSM data)
- **pin_config_reservation_aaa_prefs_gprs** (GPRS)

To configure the resource ID for the appropriate counters:

1. Open the *BRM_home***/sys/data/config/pin_config_reservation_aaa_prefs_XXX** file in a text editor.

   For our example, open the *BRM_home***/sys/data/config/pin_config_reservation_aaa_prefs_gsm_data** file, where *BRM_home* is the directory in which BRM is installed.

2. Add an entry to specify the appropriate resource ID with the entity to be rated, such as duration (**2**), volume (**3**), duration and volume (**4**), or occurrence (**8**).

   For example:

   ```
   1 PIN_FLD_RESOURCE_ID INT [0] 1000009 in REQ_MODE 4
   ```

   Here, a noncurrency resource, (Megabytes Used) with the resource ID **100009** is associated with a volume-based request (**REQ_MODE 4**).

3. Save and close the file.

For more information, see "Editing the Event Notification List" in *BRM Developer's Guide*.

Loading Reservation Preferences for Policy-Driven Charging

To load the reservation preferences list for policy-driven charging:

Copyright © 2025, 2026, Oracle and/or its affiliates.

1. Go to the *BRM_home*/**sys/data/config** directory.
2. Use the following command to run the **load_config_reservation_aaa_prefs** utility:

   ```
   load_config_reservation_aaa_prefs -d -v load_config_reservation_aaa_prefs_XXX
   ```

   where:

   - **-d** creates a log file for debugging purposes.
   - **-v** displays information about successful or failed processing as the utility runs.
   - *load_config_reservation_aaa_prefs_XXX* is the reservation configuration file.

   For example:

   ```
   load_config_reservation_aaa_prefs -d -v pin_config_reservation_aaa_prefs_gsm_data
   ```
3. To verify that the reservation preferences were loaded, display the **/config/reserve** object by using Object Browser or the **robj** command with the **testnap** utility.
4. Stop and restart the CM.

Updating the Connection Manager for Policy-Driven Charging

If you use offer profiles in price lists, ensure that you update the CM configuration file, *BRM_home*/**sys/cm/pin.conf**, for the following:

- Configure the memory required for offer profile cache. See "About Configuring the Required Memory for Offer Profiles Cache".
- Reference the system environment variables in the **pin.conf** file. See "About Referencing Offer Profile Related Variables in pin.conf Files".

About Configuring the Required Memory for Offer Profiles Cache

The offer profile cache called **fm_offer_profile_cache** is loaded with all **/offer_profile** objects at the initialization of the **fm_offer_profile** library. The **fm_offer_profile_cache** cache stores the last updated timestamp along with each **/offer_profile** object.

> **Note**
>
> Restart the CM after you make any change to the offer profile configuration entry in the **pin.conf** file.

To update the **pin.conf** file for the **fm_offer_profile_cache** entry, see "Updating the pin.conf Configuration File for Offer Profiles".

About Referencing Offer Profile Related Variables in pin.conf Files

You can reference the system environment variable for offer profiles in **pin.conf** configuration file to facilitate future migration of the **pin.conf** file to BRM implementations on other platforms. See "System Administration pin.conf Reference" in *BRM System Administrator's Guide*.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **⚠ Caution**
>
> Restart the CM after you make any change to the **/offer_profile** object.

To update the **pin.conf** file for this entry, see "Updating the pin.conf Configuration File for Offer Profiles".

Updating the pin.conf Configuration File for Offer Profiles

To update the CM **pin.conf** file:

1. Open the *BRM_home***/sys/cm/pin.conf** file in a text editor.
2. Configure the required memory for the offer profile cache using appropriate values for *number_of_entries*, *cache_size*, *hash_size* in the following command. Add this line, if necessary.

   ```
   - cm_cache fm_offer_profile_cache number_of_entries cache_size hash_size
   ```

   For example:

   ```
   - cm_cache fm_offer_profile_cache 20, 102400, 13
   ```

   For more information, see "Improving Performance for Loading Large Product Offerings" in *BRM System Administrator's Guide*.
3. Verify that the following command to reference the offer profile environmental variable from within the **pin.conf** file is present in the file:

   ```
   - cm fm_module ${PIN_HOME}/lib/fm_offer_profile${LIBRARYEXTENSION}
   fm_offer_profile_config fm_offer_profile_init pin
   ```

   Uncomment this command if it is commented, or add this command if it is not present in the **pin.conf** file.

   For more information, see "Preparing for Platform Migration by Using Variables in pin.conf Files" in *BRM System Administrator's Guide*.
4. Save the file.
5. Stop and restart the CM.

For more information, see "Improving Connection Manager Performance" in *BRM System Administrator's Guide*.

Customizing Information Received from BRM

BRM uses the following opcodes to support offer profile threshold notifications:

- PCM_OP_BAL_APPLY_MULTI_BAL_IMPACTS
- PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS
- PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD
- PCM_OP_TCF_AAA_POL_GET_SUBSCRIBER_PROFILE
- PCM_OP_ACT_POL_EVENT_NOTIFY

This section describes these opcodes and how you can customize them, if necessary.

Copyright © 2025, 2026, Oracle and/or its affiliates.

About the PCM_OP_BAL_APPLY_MULTI_BAL_IMPACTS Opcode

BRM uses the PCM_OP_BAL_ APPLY_MULTI_BAL_IMPACTS opcode to process both offline and online charging events. This opcode retrieves the current balance and calls PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS policy opcode.

About the PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS Policy Opcode

The PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS policy opcode sets up an offer profile threshold breach notification when it encounters a threshold breach in a customer's usage of a resource defined in an offer profile (associated with the customer's purchased rate plan).

When the policy opcode is called by BRM as part of an authorization or reauthorization of a subscriber's request to use a service, it sets up an in-session notification for the offer profile threshold breach it encounters for the customer's usage of the resource. When the policy opcode is called by BRM during billing or accounts receivable operations, it sets up an out-of-session notification for the offer profile threshold breach it encounters for the customer's usage of the resource.

The PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS policy opcode does not handle prerated events.

When you start (or restart) Connection Manager, BRM caches the set of unique resource IDs that you have configured in your offer profiles. The PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS policy opcode uses this cache of resource IDs as the initial filter when applying the balance impact of each resource to retrieve the offer profile information and determine if an offer profile threshold notification is necessary. If a resource is not present in the cached entries, this policy opcode does not retrieve the offer profile information for that resource. Consequently, it will not set up an offer profile threshold breach notification, even if such a notification is necessary.

> **Note**
>
> If you introduce a new resource ID, stop and restart the CM.

How Balances Are Processed for Out-of-Session Notifications

For out-of-session rating events, the PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS policy opcode uses the offer profile associated with the product or discount that granted this resource (called the grantor object).

The policy opcode retrieves all the services associated with the balance group and records an offer profile threshold breach notification event for all the services where an offer profile threshold was breached.

How Balances Are Processed for Notifications for Prepaid Sessions

For in-session rating events, the PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS policy opcode sets up offer profile threshold breach notifications for the subscriber only. It does not take resource sharing into consideration when it sets up these notifications.

Copyright © 2025, 2026, Oracle and/or its affiliates.

This policy retrieves the offer profile associated with the purchased product or discount in the balance impact data it received from the rating process. It uses the offer profile information, the subscriber's current balances and the consumed reservation amount for the resource to determine whether an offer profile threshold was breached. If so, it records an offer profile threshold breach notification event.

Customizing Information in Offer Profile Threshold Breach Notifications

See "Sample Threshold Breach Notification" for the default information in an offer profile threshold breach notification. You can customize the PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS policy opcode to retrieve other information on the resource usage.

About the PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD Opcode

The PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD opcode uses offer profiles passed in the input flist to retrieve the dynamic information for a given service identifier and resource ID.

BRM calls this opcode to perform the following operations:

- Determining Whether Balance Impacts Trigger Notifications
- Readjusting Quotas for Requests to Update and Authorize
- Readjusting Quotas for Other AAA Requests

For more information on the PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD opcode, see *Opcode Flist Reference*.

Determining Whether Balance Impacts Trigger Notifications

To determine whether any threshold breach notifications result from balance impacts due to the resource usages, BRM calls the PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD opcode with the balances (without current impact) stored in the BALANCES array of opcode's input flist.

The PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD opcode does the following:

1. Calculates the sum of the current balance and the consumed reservation from the BALANCES array.
2. Locates old used quota and the last threshold notification.
3. Calls the PCM_OP_BAL_GET_PREPAID_BALANCES opcode with the resource ID to compute the new used amount.

   The PCM_OP_BAL_GET_PREPAID_BALANCES opcode returns the sum of the updated current balance and consumed reservation as the new used amount.
4. Locates the status label, offer profile name and policy label name for the tier with the nearest threshold. To do so, it employs the new used amount and the range from the last threshold notification.
5. Calculates the offset to the next threshold as the difference between the new used amount and the next threshold on that tier.

Copyright © 2025, 2026, Oracle and/or its affiliates.

6. Sets the PIN_FLD_INDICATOR field in the output to **0** if there is a breach. And returns the calculated offset to the next threshold.

Readjusting Quotas for Requests to Update and Authorize

BRM calls the PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD opcode with the additional quota reported by the network in the QUANTITY field of the input list.

The PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD opcode does the following:

1. Calls the PCM_OP_BAL_GET_PREPAID_BALANCES opcode with the resource ID to retrieve the current balance and consumed reservation.
2. Calculates the total used as the sum of the current balance, the consumed reservation, and the additional quantity reported by the network.
3. Locates the status label, offer profile name, and policy label name for the tier with the nearest threshold with reference to the total used amount.
4. Calculates the offset to the next threshold as the difference between the new used amount and the next threshold on that tier.
5. Returns the calculated offset to the next threshold

Readjusting Quotas for Other AAA Requests

When called to readjust the quota for AAA requests other than the update and reauthorize request, the PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD opcode does the following:

1. Retrieves the current balance and consumed reservation by calling the PCM_OP_BAL_GET_PREPAID_BALANCES opcode with the resource ID.
2. Calculates the total used as the sum of the current balance and the consumed reservation.
3. Locates the status label, offer profile name and policy label name for the tier with the nearest threshold with reference to the total used amount.
4. Calculates the offset to the next threshold as the difference between the new used amount and the next threshold on that tier.
5. Returns the calculated offset to the next threshold.

About the PCM_OP_TCF_AAA_GET_SUBSCRIBER_PROFILE Opcode

The PCM_OP_TCF_AAA_GET_SUBSCRIBER_PROFILE opcode returns the following information from the database:

- All the information associated with a subscriber's mobile station ID (MSID)
- Specific service only for a given object identifier (POID)

The PCM_OP_TCF_AAA_GET_SUBSCRIBER_PROFILE opcode uses the internal PCM_OP_OFFER_PROFILE_GET_OFFER_PROFILE opcode to fetch the necessary information. It returns the information in the PIN_FLD_RESULTS array with PIN_FLD_ACCOUNT containing information associated with the account and PIN_FLD_SERVICE_INFO containing information on the services.

The opcode returns the following information based on the value you input for the PIN_FLD_MODE entry when you call this opcode:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Static information on the resource used by a subscriber such as offer profiles and subscriber's preference data (PIN_FLD_MODE is **1**)
- Dynamic information such as current balances for the various resources (PIN_FLD_MODE is **2**)
- Both types of information (PIN_FLD_MODE is **3**)

For more information on the PCM_OP_TCF_AAA_GET_SUBSCRIBER_PROFILE opcode, see *Opcode Flist Reference*.

Customizing Information Received from BRM

Use the PCM_OP_TCF_AAA_POL_GET_SUBSCRIBER_PROFILE policy opcode to customize the information your network policy controlling application retrieves from BRM.

By default, PCM_OP_TCF_AAA_POL_GET_SUBSCRIBER_PROFILE policy opcode calls the PCM_OP_TCF_AAA_GET_SUBSCRIBER_PROFILE opcode and returns the output from that opcode.

For more information on the PCM_OP_TCF_AAA_POL_GET_SUBSCRIBER_PROFILE policy opcode, see *Opcode Flist Reference*.

About the PCM_OP_ACT_POL_EVENT_NOTIFY Policy Opcode

When a product or discount associated with an **offer_profile** is purchased, canceled, or modified, the PCM_OP_ACT_POL_EVENT_NOTIFY policy opcode calls the PCM_OP_OFFER_PROFILE_GET_OFFER_PROFILE opcode to retrieve the associated offer profile data and adds it to the event notification. The PCM_OP_OFFER_PROFILE_GET_OFFER_PROFILE opcode is internal to BRM. See "About the PCM_OP_TCF_AAA_GET_SUBSCRIBER_PROFILE Opcode" for more information.

For more information on the PCM_OP_ACT_POL_EVENT_NOTIFY policy opcode, see *Opcode Flist Reference*.

Customizing Information in Event Notifications for Policy-Driven Charging

Use the PCM_OP_ACT_POL_EVENT_NOTIFY policy opcode to customize the event notification information associated with policy-driven charging that you want to receive from BRM.

Managing Offer Profiles with Opcodes

You can manage offer profile data using price list opcodes and offer profile opcodes.

Managing Offer Profile Data with Price List Opcodes

Use the PCM_OP_PRICE_SET_PRICE_LIST and PCM_OP_PRICE_GET_PRICE_LIST opcodes to manage your offer profiles.

Creating, Modifying, and Deleting Offer Profiles with PCM_OP_PRICE_SET_PRICE_LIST

The PCM_OP_PRICE_SET_PRICE_LIST opcode searches the database for each offer profile in the PIN_FLD_OFFER_PROFILES array. To create or modify offer profiles in the database, input the **/offer_profile** objects in the PIN_FLD_OFFER_PROFILES array. To delete an offer

Copyright © 2025, 2026, Oracle and/or its affiliates.

profile from the database, input the **/offer_profile** object in the PIN_FLD_OFFER_PROFILES array and specify the PIN_FLD_DELETE_FLAG entry in the input flist.

The PCM_OP_PRICE_SET_PRICE_LIST opcode searches the database for each offer profile in the PIN_FLD_OFFER_PROFILES array and does one of the following:

- If the offer profile is not found, the opcode creates and stores an **/offer_profile** object with this data.
- If the offer profile is found, the opcode retrieves that **/offer_profile** object and does one of the following:
  - If the PIN_FLD_DELETE_FLAG entry is not present in the input flist, the opcode updates **/offer_profile**, and stores the updated object.
  - If the PIN_FLD_DELETE_FLAG entry is present in the input flist, the opcode deletes the **/offer_profile** object from the database.

See "Managing Price Lists" for more information.

Retrieving Offer Profiles with PCM_OP_PRICE_GET_PRICE_LIST

Use the PCM_OP_PRICE_GET_PRICE_LIST opcode to retrieve the contents of an **/offer_profile** object from the database. The PIN_FLD_OFFER_PROFILES array in the output contains the **/offer_profile** objects.

See "Managing Price Lists" for more information.

Managing Offer Profiles with Offer Profile Opcodes

Manage offer profile data using the following:

- PCM_OP_OFFER_PROFILE_SET_OFFER_PROFILE. See "About the PCM_OP_OFFER_PROFILE_SET_OFFER_PROFILE Opcode".
- PCM_OP_TCF_AAA_POL_GET_SUBSCRIBER_PROFILE. See "About the PCM_OP_TCF_AAA_GET_SUBSCRIBER_PROFILE Opcode".

About the PCM_OP_OFFER_PROFILE_SET_OFFER_PROFILE Opcode

Use the PCM_OP_OFFER_PROFILE_SET_OFFER_PROFILE opcode to create, modify, and delete offer profiles.

Input the offer profiles in the PIN_FLD_OFFER_PROFILES array. To delete an offer profile, specify the PIN_FLD_DELETE_FLAG entry for the offer profile in the input flist.

The opcode searches the database for the offer profiles in the PIN_FLD_OFFER_PROFILES array and does one of the following:

- If the offer profile is not found, the opcode creates and stores an **/offer_profile** object with this data.
- If the offer profile is found, the opcode retrieves that **/offer_profile** object and
  - If the PIN_FLD_DELETE_FLAG entry is not present in the input flist, it updates **/offer_profile**, and stores the updated object.
  - If the PIN_FLD_DELETE_FLAG entry is present in the input flist, it deletes the **/offer_profile** object.

For more information on the PCM_OP_OFFER_PROFILE_SET_OFFER_PROFILE opcode, see *BRM Developer's Reference*.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Customizing Subscriber Preferences Data for Policy-Driven Charging

BRM stores the default entries for subscriber preferences associated with policy-driven charging in the **config_subscriber_preferences_map.xml** file.

To customize subscriber preferences data for policy-driven charging:

1. Open the *BRM_home*/**sys/data/config/config_subscriber_preferences_map.xml** configuration file in a text editor.
2. Edit the file as necessary. You can add other subscriber preferences.
   The elements in the XML file are described in the discussion on subscriber preferences in *BRM Telco Integration*.
3. Save and close the file.

Customer Center uses the configurations in the **/config/subscriber_preferences_map** object to dynamically list the preferences that a subscriber can configure. You can customize the information as necessary.

Customizing Business Events Associated with Policy-Driven Charging

If you customize business events associated with policy-driven charging, include your definitions of these events in the **payloadconfig_ipc.xml** configuration file. For more information, see "About Publishing Additional Business Events" in *BRM Developer's Guide*.

Sample Threshold Breach Notification

Example 8-3 shows the contents of a sample offer profile threshold breach notification.

Example 8-3 Sample Offer Profile Threshold Breach Notification

```
PIN_EVENT_TY('ThresholdBreach', '<?xml version="1.0"?>
<ThresholdBreach Version="1.0">
   <service_obj>0.0.0.1 /service/telco 223789 7</service_obj>
   <account_obj>0.0.0.1 /account 10 1 </account_obj>
   <login>test</login>
   <alias_list>
      <name>694-20120607-231907-3-9178 --server.example.com</name>
   </alias_list>
   <resource_list>
      <resource_name>Megabytes Used</resource_name>
      <resource_id>100009</resource_id>
      <current_bal>35.28</current_bal>
      <unit>0</unit>
      <threshold_breach >
     <status_label>HIGH_QOS</status_label>
     <delta_to_next_threshold>14.72</delta_to_next_threshold>
     <policy_label>Fair Usage</policy_label>
     <offer_profile_name>platinum</offer_profile_name>
     </threshold_breach>
   </resource_list>
</ThresholdBreach>
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

9

Working with Promotions

This chapter describes how to implement promotions in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

- About Promotions
- Applying Promotions On Special Dates
- Applying Promotions for Special Events or Actions

About Promotions

You can apply promotions to your customers, such as by granting 100 free minutes, 2 GB of free data, or 2 free movie downloads, in the following situations:

- On special dates, such as a customer's birthday or membership anniversary. See "Applying Promotions On Special Dates".
- Events or actions that occur in BRM, such as a customer purchasing a product or successfully topping up an account balance. See "Applying Promotions for Special Events or Actions".

Applying Promotions On Special Dates

You specify whether customers qualify for a promotion based on a special date by adding a promotion tag and date to their account profile. For example, to apply a promotion on a customer's birthday, you could add the promotion tag BIRTHDAY and promotion date of June 15. On June 15, BRM could grant the customer 100 free minutes, 5 Gigabytes of free data, or so on.

BRM applies the promotions to your customers' accounts when the **pin_apply_promotion** utility is run. The utility searches through each account's profile for a promotion date that occurs on or before the current date. When it finds a match, it looks up the promotion event associated with the promotion tag and then rates the promotion event, which in turn grants the free resources.

To configure BRM to apply promotions on special dates:

1. When you create an account or modify an account profile, include a promotion tag and date.

   The promotion data is stored in an account's **/profile/specialdates** object.

2. Create a promotion event (**/event/promotion/** subclass) for each promotion that your company supports and then load the promotion events into the BRM database.

   For example, you could create an **/event/promotion/birthday** storable class for applying birthday promotions. BRM uses this event for rating and applying the promotion to your customers' accounts.

   For more information about:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Creating custom events, see "Creating Service and Event Storable Classes" in *BRM Developer's Guide*.
- Loading custom events into the BRM database, see "load_event_map".

3. Map your promotion tags to the promotion events you created in step 2. See "Mapping Promotion Tags to Promotion Events".

   The mappings are stored in the **/config/event_promo_tag_map** object.

4. Define the amount and type of free resources that BRM needs to grant your customers when rating each promotion event. To do so, define how to apply credits for each promotion event in your system-level or account-level charge offers and rates.
5. Optionally, customize how promotions are applied. See "Customizing Special Date, Event, or Action Promotions" in *BRM Opcode Guide*.
6. Run the **pin_apply_promotion** utility on a daily basis. See "pin_apply_promotion".

Mapping Promotion Tags to Promotion Events

To map promotion tags to promotion events:

1. Open the *BRM_home***/sys/data/config/config_event_promo_tag_map.xml** file in a text editor.

   The **config_event_promo_tag_map.xml** file contains sample mappings, but you can modify them to meet your business needs.

2. Add an **<EVENT_MAP>** array element for each promotion tag that you want mapped to a promotion event. Table 9-1 describes the elements in **<EVENT_MAP>**.

Table 9-1 Elements in config_event_promo_tag_map.xml

Element	Description
<PROMO_EVENT_TYPE>	The promotion event to rate.
<PROMO_TAG>	The name of the promotion tag.

The following sample content maps the **BIRTHDAY** promotion tag to the **/event/promotion/birthday** promotion event:

```
<ConfigObject>
   <DESCR>Event Promo Tag Map Configuration</DESCR>
   <HOSTNAME>-</HOSTNAME>
   <NAME>Event Promo Tag Map</NAME>
   <PROGRAM_NAME>load_config</PROGRAM_NAME>
   <EVENT_MAP elem="0">
      <PROMO_EVENT_TYPE>/event/promotion/birthday</PROMO_EVENT_TYPE>
      <PROMO_TAG>BIRTHDAY</PROMO_TAG>
   </EVENT_MAP>
</ConfigObject>
```

3. Save and close the file.
4. Edit the **load_config** utility's configuration file (*BRM_home***/apps/load_config/pin.conf**) to specify how to connect to the BRM database.

Copyright © 2025, 2026, Oracle and/or its affiliates.

5. Load the **config_event_promo_map.xml** file into the BRM database:

   ```
   load_config -d -v BRM_home/sys/data/config/config_event_promo_tag_map.xml
   ```

   See "load_config" in *BRM Developer's Guide* for more information.

6. Stop and restart the Connection Manager (CM).
7. Verify that the mappings were loaded properly by running the following command:

   ```
   load_config -w "event_promo_tag_map" config_event_promo_tag_map_out.xml
   ```

Applying Promotions for Special Events or Actions

You can configure BRM to apply promotions for special events or actions by associating BRM event types with your promotions. For example, you could specify that all **/event/billing/product/action/purchase** events qualify for a promotion. In this case, every time customers purchase a new product, BRM could grant them 5 gigabytes of free data, 1 free movie download, or so on.

You implement these types of promotions by using the BRM event notification system. When a specified BRM event occurs, the event notification system triggers a call to the PCM_OP_SUBSCRIPTION_HANDLE_PROMO_EVENT opcode which in turn looks up the promotion event associated with the BRM event. For example, it could look up that **/event/billing/product/action/purchase** events are associated with the **/event/promotion/subscribe_offer** promotion event. BRM then rates the promotion event, which grants the free resources.

To configure BRM to apply promotions for special events or actions:

1. Create a promotion event (**/event/promotion/** subclass) for each promotion that your company supports and then load the promotion events into the BRM database.

   For example, you could create an **/event/promotion/subscribe_offer** storable class. BRM uses this event for rating and applying the promotion to your customers' accounts.

   For more information about:

   - Creating custom events, see "Creating Service and Event Storable Classes" in *BRM Developer's Guide*.
   - Loading custom events into the BRM database, see "load_event_map".

2. Configure the event notification system to trigger calls to the PCM_OP_SUBSCRIPTION_HANDLE_PROMO_EVENT opcode when a BRM event occurs. See "Configuring Event Notification for Promotions".
3. Map BRM events to the promotion events you created in step 1. See "Mapping BRM Events to Promotion Events".

   The mappings are stored in the **/config/event_promo_map** object.

4. Define the amount and type of free resources that BRM needs to grant your customers when rating each promotion event. To do so, define how to apply credits for each promotion event in your system-level or account-level charge offers and rates.
5. Optionally, customize how promotions are applied. See "Customizing Special Date, Event, or Action Promotions" in *BRM Opcode Guide*.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Configuring Event Notification for Promotions

To configure event notification for promotions, you must map each BRM event to which you want to apply promotions to the PCM_OP_SUBSCRIPTION_HANDLE_PROMO_EVENT opcode. For more information about event notification, see "Using Event Notification" in *BRM Developer's Guide*.

To configure event notification for promotions:

1. Open the *BRM_home***/sys/data/config/pin_notify** file in a text editor.
2. Add an entry for each BRM event that should trigger calls to opcode **9075** (the number for the PCM_OP_SUBSCRIPTION_HANDLE_PROMO_EVENT opcode).

   For example, to trigger calls to PCM_OP_SUBSCRIPTION_HANDLE_PROMO_EVENT whenever customers successfully top up their account balances, purchase a new product, or create an account profile, you would add the following lines:

   ```
   # Event Notification for Promotions
   9075    0       /event/billing/topup
   9075    0       /event/billing/product/action/purchase
   9075    0       /event/notification/profile/create
   ```

3. Save and close the file.
4. Load your **pin_notify** file into the BRM database by using the **load_pin_notify** utility:

   load_pin_notify *BRM_home*/**sys/data/config/pin_notify**

   See "load_pin_notify" in *BRM Managing Customers* for more information.
5. Stop and restart the Connection Manager (CM).

Mapping BRM Events to Promotion Events

To map BRM events to promotion events:

1. Open the *BRM_home***/sys/data/config/config_event_promo_map.xml** file in a text editor.

   The **config_event_promo_map.xml** file contains sample mappings, but you can modify them to meet your business needs.
2. Add an **<EVENT_MAP>** array element for each BRM event that you want mapped to a promotion event. Table 9-2 describes the elements in **<EVENT_MAP>**.

Table 9-2 Elements in config_event_promo_map.xml

Element	Description
<EVENT_TYPE>	The BRM event.
<FIELD_NUM>	The field number defined in the *BRM_home***/include/pin_flds.h** file. For example, you would use field number **601** for PIN_FLD_LABEL. **Note**: This applies to fields with a data type of PIN_FLDT_STR only.
<FIELD_VALUE>	The value of the field specified in FIELD_NUM.
<PROMO_EVENT_TYPE>	The promotion event.

Copyright © 2025, 2026, Oracle and/or its affiliates.

The following sample content maps the **/event/billing/topup** event to the **/event/promotion/topup** promotion event:

```
<ConfigObject>
   <DESCR>Event Promo Map Configuration</DESCR>
   <HOSTNAME>-</HOSTNAME>
   <NAME>Event Promo Map</NAME>
   <PROGRAM_NAME>load_config</PROGRAM_NAME>
   <EVENT_MAP elem="0">
      <EVENT_TYPE>/event/billing/topup</EVENT_TYPE>
      <FIELD_NUM>55</FIELD_NUM>
      <FIELD_VALUE>Topup Event (acct)</FIELD_VALUE>
      <PROMO_EVENT_TYPE>/event/promotion/topup</PROMO_EVENT_TYPE>
   </EVENT_MAP>
</ConfigObject>
```

3. Save and close the file.
4. Edit the **load_config** utility's configuration file (*BRM_home***/apps/load_config/pin.conf**) to specify how to connect to the BRM database.
5. Load the **config_event_promo_map.xml** file into the BRM database:

   ```
   load_config -d -v BRM_home/sys/data/config/config_event_promo_map.xml
   ```

   See "load_config" in *BRM Developer's Guide* for more information.
6. Stop and restart the Connection Manager (CM).
7. Verify that the mappings were loaded properly by running the following command:

   ```
   load_config -w "event_promo_map" config_event_promo_map_out.xml
   ```


Copyright © 2025, 2026, Oracle and/or its affiliates.

10

Working with Provisioning Tags

This chapter describes how to implement Oracle Communications Billing and Revenue Management (BRM) provisioning tags if you are using Pricing Center.

Topics in this document:

- About Provisioning Tags
- Using the Provisioning Tag Framework
- Default Provisioning Tag for Policy-Driven Charging

About Provisioning Tags

Provisioning tags implement extended rating attributes (ERAs) or other user-defined attributes that can enable rating or discounting to vary for a service. For telco services, you can also use provisioning tags to provision supplementary services and service extensions.

An offer profile and the provisioning tag for the associated product or discount use the same name and that name must be unique. If you create an offer profile to associate with an existing product or discount, use the provisioning tag to name the offer profile. If you configure a new product or discount around an existing offer profile, use the appropriate offer profile name as the provisioning tag for the product or discount.

You can use one of the following methods to define provisioning tags, depending on what the tag is for:

- **Provisioning tag framework**: You create provisioning tags by defining them in an XML file that is loaded into a configuration object in BRM. You can create provisioning tags for any service or for an account.

  You also might need to add the tag to the provisioning policy source file and compile the file.

  See "Using the Provisioning Tag Framework".

  > **Note**
  >
  > – For telco services, use Services Framework Manager to define provisioning tags in most cases. You must use Services Framework Manager if the provisioning tag creates supplementary services or service extensions. See "About Provisioning Tags for Telco Services" in *BRM Provisioning Services*.
  >
  > – For non-telco services, the provisioning tag framework is the recommended method of creating provisioning tags.

- **Provisioning Tags application in Pricing Center**: For telco services only, you can create service-level provisioning tags with this application. You can include service-level ERAs, supplementary services, and service extensions in the tags.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> You can create provisioning tags that include existing ERAs only.

- **Telco tag text file**: For telco services only, you can create service-level provisioning tags by defining them in a text file and then loading the file into BRM. You can include supplementary services and ERAs in the tags.

  You can also create account-level ERAs in this same file.

  See "Defining Provisioning Tags for Telco Services by Using the pin_telco_tags File" in *BRM Provisioning Services*.

- **Provisioning policy source file**: You can define product provisioning tags by editing and compiling the policy file that is the source file for all provisioning operations.

> **Note**
>
> The provisioning tag framework is the preferable method for creating provisioning tags.

See "Using a Policy Source File to Set Up Provisioning".

Defining Provisioning Tags for Telco Services by Using the pin_telco_tags file

This section describes defining provisioning tags through the **pin_telco_tags** file for a specific telco service. For example, you use **pin_telco_tags_gsm** for GSM provisioning tags.

You can include service-level ERAs, supplementary services, and service extensions in a provisioning tag defined in a **pin_telco_tags** file.

You can use the Provisioning Tags application in Pricing Center instead of the **pin_telco_tags** file to define provisioning tags for telco services. But you cannot create custom ERAs using Provisioning Tags. For information, see Provisioning Tags Help.

To define provisioning tags using the **pin_telco_tags** file:

- Configure provisioning tags in the **pin_telco_tags**_*service* file. See "Configuring Provisioning Tags in the pin_telco_tags File".
- Load the **pin_telco_tags**_*service* file into the BRM database with the **load_pin_telco_tags** utility. See "Loading the pin_telco_tags File ".

You can also define account-level ERAs in the **pin_telco_tags** file. See "Defining Account-Level ERAs in the pin_telco_tags File ".

Configuring Provisioning Tags in the pin_telco_tags File

To configure a provisioning tag in the **pin_telco_tags** file:

1. Open the **pin_telco_tags**_*service* file. For example, use **pin_telco_tags_gsm** for GSM services.

   The default **pin_telco_tags** files are in *BRM_home*/**sys/data/config**. They include examples and instructions.

Copyright © 2025, 2026, Oracle and/or its affiliates.

2. Use this syntax to add a provisioning tag:

```
provisioning_tag  "Class" "ProvTag" "PTDescription" "DelayedProvReqd"
service_extn      "Extension Type Name" "Extension Value"
features          "One Or More Feature Name String Values"
service_era       "ServiceERA" "StringIdERA" "StringIdServiceERADesc" "ProvBool" "ERALabel"
```

   Enter each value in quotation marks.

A provisioning tag can be any combination of service extensions, features, and service-level ERAs. You do not need to include all three types of data in a tag.

Table 10-1 describes the provisioning tag syntax.

Table 10-1 Provisioning Tag Syntax

Tag Element	Value	Description
provisioning_tag	Class	The object that stores the tag. For example: **"/config/telco/gsm/telephony"**
provisioning_tag	ProvTag	The name of the provisioning tag. For example: **"DataPremium"**
provisioning_tag	PTDescription	The description of the provisioning tag. For example: **"Premium Data Service"**
provisioning_tag	DelayedProvReqd	Whether the tag is unprovisioned when the product containing the tag is canceled. The possible values are: • **"y"** specifies that cancellation triggers unprovisioning. In most cases, use this setting. • **"n"** specifies that cancellation does not trigger unprovisioning. Use this setting to leave a customer's service configuration unchanged. For example, you might want to leave a voice mailbox intact.
service_extn	Extension Type Name	The type of service extension. For example: **"BEARER_SERVICE"**
service_extn	Extension Value	The code for a GSM bearer service or other service extension. For example: **"B46"** Codes are defined in the GSM specification. You must use the exact code that the network requires.
features	One or More Feature Name String Values	The GSM supplementary services that are provisioned when this product is purchased. The services are entered as codes, in one line. For example: **"CLIP" "CW"** These codes are defined in the GSM specification. You must use the exact code that the network requires.
service_era	ServiceERA	The service ERA code. For example: **"FRIENDS_FAMILY"**
service_era	StringIdERA StringIdServiceERADesc	The IDs for the ERA name and description. For example: **"12" "13"** You define a localized name and description for these IDs in the **era_descr.***locale*file. These names and descriptions appear in Customer Center.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 10-1 (Cont.) Provisioning Tag Syntax

Tag Element	Value	Description
service_era	ProvBool	Whether or not provisioning is required. The possible values are: • **"y"** specifies that provisioning is required. • **"n"** specifies that provisioning is not required.
service_era	ERALabel	The name of a list within the ERA. An ERA can have one or more lists. For example: **"MYFRIENDS"** **Note:** You cannot localize the ERA label. You cannot have duplicate label names associated with the same ERA code.

This example shows a provisioning tag for a telephony product that includes a bearer service, call waiting and voice mailbox supplementary services, and friends and family service-level ERAs:

```
# Standard Telephony Package
provisioning_tag  "/config/telco/gsm/telephony"  "Voice Standard" "Voice Standard
Service package with Called ID, Call Waiting, Voice mail and Friends and Family" "y"
service_extn      "BEARER_SERVICE" "T11"
features          "CLIP" "CW" "VMBOX"
service_era       "FRIENDS_FAMILY" 12 13 "n" "MYFRIENDS"
service_era       "FRIENDS_FAMILY" 12 13 "n" "MYFAMILY"
```

Loading the pin_telco_tags File

Run the **load_pin_telco_tags** utility to load the contents of the **pin_telco_tags**_*service* file: for example, the **pin_telco_tags_gsm** file: into the BRM database. This utility creates or updates **/config/telco/***service* and **/config/account_era** objects.

> **Note**
>
> - By default, the **load_pin_telco_tags** utility appends telco provisioning tags and account-level ERAs to the BRM database. But if you use the **-x** parameter, this utility overwrites existing telco provisioning tags and account-level ERAs. Do not use the **-x** parameter unless you are certain you want to overwrite existing objects.
> - The **load_pin_telco_tags** utility requires a configuration file. See "Connecting BRM Utilities" in *BRM System Administrator's Guide*.
> - You cannot create ERAs for individual brands. All ERAs can be used by any brand.

To load the **pin_telco_tags** file:

1. Edit the **pin_telco_tags**_*service* file to add the custom account-level ERAs. The default **pin_telco_tags_gsm** file in *BRM_home***/sys/data/config** includes examples and instructions.
2. Save the **pin_telco_tags**_*service* file.
3. Use the following command to run the **load_pin_telco_tags** utility:

   `load_pin_telco_tags pin_telco_tags`_*service*

Copyright © 2025, 2026, Oracle and/or its affiliates.

4. Restart the Connection Manager (CM).
5. Restart Pricing Center.

To verify that the account ERAs were loaded, you can display the **/config** objects by using the Object Browser or use the **robj** command with the **testnap** utility.

Defining Account-Level ERAs in the pin_telco_tags File

Account-level ERAs apply to any activity in an account no matter which service is involved.

You define account-level ERAs in any **pin_telco_tags**_*service* file. You can use the define account-level ERAs in the same file you used for telco provisioning tags.

> **Note**
>
> You cannot use the Provisioning Tags application for account-level ERAs.

To define an account-level ERA:

1. Open a **pin_telco_tags**_*service* file. You can define an account-level ERA in any **pin_telco_tags** file, because they are not associated with a particular service.

   The default **pin_telco_tags** files are in *BRM_home***/sys/data/config**. They include examples and instructions.

   Account-level ERAs are added in a block at the end of the file.
2. Use this syntax to add an account-level ERA:

```
account_era  "AccountERA" "StringIdERA" "StringIdAccountERADesc"
```

Table 10-2 describes the account ERA syntax.

Table 10-2 Account ERA Syntax

Value	Description
AccountERA	The account ERA code. For example: **"SPECIAL_DAY"**
StringIdERA StringIdAccountERADesc	The IDs for the ERA name and description. For example: **"2" "3"** You define a localized name and description for these IDs in the **era_descr.***locale*file. These names and descriptions appear in Customer Center. See "Localizing and Customizing Strings" in *BRM Developer's Guide* for information about updating this file.

For example:

```
account_era "SPECIAL_DAY" "2" "3"
```

Deciding Which Provisioning Tag Method to Use

For telco services, you should create provisioning tags by using one of these methods. The order of preference is shown below.

1. Provisioning tag framework. See "Using the Provisioning Tag Framework".

Copyright © 2025, 2026, Oracle and/or its affiliates.

2. Provisioning Tags application in Pricing Center.
3. Telco tag text file.
4. Provisioning policy source file. See "Using a Policy Source File to Set Up Provisioning".

For non-telco services, you should create provisioning tags by using one of these methods. The order of preference is shown below:

1. Provisioning tag framework. See "Using the Provisioning Tag Framework".
2. Provisioning policy source file. See "Using a Policy Source File to Set Up Provisioning".

Using the Provisioning Tag Framework

You can create service-level or account-level provisioning tags using the provisioning tag framework. This framework stores provisioning tags in the **/config/provisioning_tag** object.

> **Note**
>
> To create most provisioning tags for telco services, use the Provisioning Tags application in Pricing Center or the **load_pin_telco_tags** utility.

You define a provisioning tag by specifying the services to which it applies and the opcodes to run when the product or discount containing the tag is purchased or canceled. You might need to create custom opcodes for some provisioning tags.

To create a provisioning tag using this framework, do the following:

- Configure the provisioning tag in the **pin_config_provisioning_tags.xml** file. See "Configuring Provisioning Tags ".
- Load the provisioning tag configuration into the **/config/provisioning_tag** object with the **load_config_provisioning_tags** utility. "Loading Provisioning Tag Configurations".
- Add the tag to the provisioning policy source file and compile the file. You do this if you will include the provisioning tag in a product and if the tag uses a service associated with the __DEFAULT__ provisioning tag. See "Modifying and Compiling the Provisioning Policy Source File".

Configuring Provisioning Tags

To create provisioning tags, you configure the provisioning tags configuration file, **pin_config_provisioning_tags.xml**. This file is located in the *BRM_home*/**sys/data/config** directory.

To define a provisioning tag in this configuration file, you specify the following:

- A unique name for the provisioning tag.

  > **Note**
  >
  > The provisioning tag for a product or discount must be the name of the offer profile with which the product or discount is associated.

- For a service-level tag, the permitted services.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- For an account-level tag, **/account**.
- The name and number of each opcode to run when the product or discount containing the provisioning tag is purchased or canceled. These opcodes contain the business logic to perform the actual provisioning, such as creating a profile.
- Parameters that specify the fields to be added to each opcode's input flist and the value for each field.

> **Note**
>
> – Do not remove existing provisioning tags from **pin_config_provisioning_tags.xml** when adding new tags unless you want existing tags removed from the **/config/provisioning_tag** objects in the database.
>
> – When you load **pin_config_provisioning_tags.xml**, all existing instances of **/config/provisioning_tag** are removed from the database, so only the provisioning tags defined in the file when you load it will be in the database.

Table 10-3 lists the elements in the **pin_config_provisioning_tags.xml** file, the syntax to use for each element, and a description of how to specify each element. The syntax is based on the default version of the file.

Table 10-3 Elements in pin_config_provisioning_tags XML File

Element	Syntax	Description
BusinessConfiguration	`<BusinessConfiguration xmlns="http://www.portal.com/schemas/BusinessConfig" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.portal.com/schemas/BusinessConfig business_configuration.xsd">`	The root element common to all BRM configurations.
ProvisioningTag Configuration	`<ProvisioningTagConfiguration>`	Opens the provisioning tag configuration. Contains the ProvisioningTagList element.
ProvisioningTagList	`<ProvisioningTagList>`	Contains all the provisioning tag definitions.
ProvisioningTag	`<ProvisioningTag name= "__DEFAULT__">`	Contains the definition of a provisioning tag and specifies the tag's name.
PermittedTypes	`<PermittedTypes>/service/ip </PermittedTypes>`	Specifies a service or account that is valid for the provisioning tag. One or more of these tags can be part of a provisioning tag definition. When the API for getting a list of provisioning tags is called, the array of permitted types is looked up and only provisioning tags applicable to the specific service type are returned.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 10-3 (Cont.) Elements in pin_config_provisioning_tags XML File

Element	Syntax	Description
OpcodeList	`<OpcodeList>`	Contains the definition of one opcode. The opcodes specified for a provisioning tag contain the business logic required for provisioning when purchasing and canceling a product. A provisioning tag can include multiple opcodes.

OpcodeName	`<OpcodeName>`
	`PCM_OP_SUBSCRIPTION_POL_PURCHASE_PROD_PROVISIONING`
	`</OpcodeName>`
		Specifies the opcode's name. See "Specifying an Opcode in a Provisioning Tag to Create an ERA".

OpcodeNumber	`<OpcodeNumber>417` `</OpcodeNumber>`	Specifies the hard-coded number for an opcode. To find an opcode's number, see the opcode header files in the *BRM_home*/**include/ops** directory.
OpcodeMode	`<OpcodeMode>0</OpcodeMode>`	Indicates when the opcode should be called, as follows: • **0**: on product or discount purchase • **1**: on product or discount cancellation • **2**: on both purchase and cancellation
OpcodeParamsList	`<OpcodeParamsList>`	Defines a parameter for an opcode. You can have multiple parameters. Each parameter is a name-value pair.
OpcodeParamName	`<OpcodeParamName>PIN_FLD_POID` `</OpcodeParamName>`	Specifies a field name to be added to the input flist. If the field is part of a substruct or array, use a period to separate the substruct or array name and the field name. For example: `PIN_FLD_EVENT.PIN_FLD_ACCOUNT_OBJ`
OpcodeParamValue	`<OpcodeParamValue>$SERVICE$` `</OpcodeParamValue>`	Specifies the value of the field. For certain values that are not known until run time, you can use a variable, such as $SERVICE$. See "Variables for Parameter Values".

For an example of the XML syntax, see "Sample Provisioning Tag XML File".

Specifying an Opcode in a Provisioning Tag to Create an ERA

When defining a configuration for the provisioning tag framework, you specify one or more opcodes to perform an action, such as creating an ERA. You can specify that an opcode run

Copyright © 2025, 2026, Oracle and/or its affiliates.

when a product or discount is purchased, canceled, or both. You can use an existing opcode or design a custom opcode.

If the provisioning tag is designed to create an ERA, you can specify that the PCM_OP_SUBSCRIPTION_PROVISION_ERA opcode run at both purchase and cancellation time. This opcode creates, modifies, or deletes a profile (**/profile** object). ERAs are stored in profiles.

The actions PCM_OP_SUBSCRIPTION_PROVISION_ERA takes depends on the value specified for the PIN_FLD_ACTION parameter in the provisioning tag:

- If PIN_FLD_ACTION is **Purchase**, the opcode checks if a profile already exists. If the profile does not exist, the opcode calls PCM_OP_CUST_CREATE_PROFILE to create the profile. If the profile does exist, the opcode calls PCM_OP_CUST_MODIFY_PROFILE to add data passed in from the input flist and to increment the reference counter by 1.
- If PIN_FLD_ACTION is **Cancel**, the opcode decrements the reference counter by 1. If the counter is 0, the opcode calls PCM_OP_CUST_DELETE_PROFILE to delete the profile.

For example, you can create a friends and family ERA for a service by calling the PCM_OP_SUBSCRIPTION_PROVISION_ERA at purchase time with the parameters in Table 10-4:

Table 10-4 PCM_OP_SUBSCRIPTION_PROVISION_ERA Parameters

OpcodeParamName	OpcodeParamValue
PIN_FLD_POID	0.0.0.0 /profile/serv_extrating -1
PIN_FLD_ACCOUNT_OBJ	$ACCOUNT$
PIN_FLD_FLAGS	0
PIN_FLD_SERVICE_OBJ	$SERVICE$
PIN_FLD_STR_VAL	12, 13
PIN_FLD_NAME	FRIENDS_FAMILY
PIN_FLD_INHERITED_INFO. PIN_FLD_EXTRATING. PIN_FLD_LABEL	MYFRIENDS

These name-value pairs indicate that an ERA named FRIENDS_FAMILY and an ERA label named MYFRIENDS are created. Because both the account and service POIDs are specified, the opcode creates a service-level profile (**/profile/serv_extrating**) object. If the service POID was not specified, the opcode would create an account-level profile (**/profile/acct_extrating**) object.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> - PIN_FLD_POID is the POID, in string format, of the object the provisioning tag will create. This is converted to a POID when the opcode runs. If you are using multiple database schemas, the string is converted to the correct schema database number.
> - For a service-level profile, the POID type is **/profile/serv_extrating**, as in the previous example. For an account-level profile, the POID type is **/profile/acct_extrating**.
> - PIN_FLD_FLAGS specifies that PCM_OP_SUBSCRIPTION_PROVISION_ERA is called at purchase time.

You must include the opcode twice in a provisioning tag, once with PIN_FLD_FLAGS set to 0 and once with PIN_FLD_FLAGS set to 1, so that it runs both at purchase and cancellation time.

- PIN_FLD_STR_VAL specifies that the profile name and profile description are localized and are stored in the **/string** object under string IDs 12 and 13.
- PIN_FLD_ACCOUNT_OBJ and PIN_FLD_SERVICE_OBJ use variables. See "Variables for Parameter Values".

Variables for Parameter Values

You can use the following variables in Table 10-5 to specify certain values available only at the time the opcode is run:

Table 10-5 Run-Time Variables for Parameters

Variable	Description
$ACCOUNT$	Account POID
$SERVICE$	Service POID
$PRODUCT$	Product POID
$DISCOUNT$	Discount POID
$OFFERING$	POID of the purchased product or discount
$PROVTAG$	Provisioning tag

Default Provisioning Tag

The default **/config/provisioning_tag** object contains the __DEFAULT__ provisioning tag. This tag is defined in the default **pin_config_provisioning_tags.xml** file. The tag calls the following opcodes:

- PCM_OP_SUBSCRIPTION_POL_PURCHASE_PROD_PROVISIONING, on product purchase.
- PCM_OP_SUBSCRIPTION_POL_CANCEL_PROD_PROVISIONING, on product cancellation.

If you have customized these opcodes, they set and clear fields in **/service** objects when a product is purchased and canceled. If you have not customized these opcodes, you do not

Copyright © 2025, 2026, Oracle and/or its affiliates.

need to use them. You can specify other opcodes in the provisioning tags file to perform necessary actions.

The __DEFAULT__ provisioning tag is always called for specified services when a product is purchased or canceled, so you do not need to include these opcodes in any other provisioning tags you define. See the default **pin_config_provisioning_tags.xml** file for the list of services.

Using Custom Opcodes

You can create custom opcodes to perform actions not supported by existing opcodes and specify the custom opcodes in the provisioning tags configuration file. For example, you can write an opcode to add fields to a **/service** object.

Configuring Provisioning Tags for Policy-Driven Charging

Configure the resources that BRM should track by creating provisioning tags for your services. Each provisioning tag contains the types of resources valid for that tag and information on the policy attributes which are used in provisioning policy for that resource.

Policy attributes are the configured characteristics associated with a provisioning policy. They contain the information you must provide to the specific opcode which is to process changes to the product or discount associated with that provisioning tag (for example, an offer profile threshold breach notification is required and/or the language in which the subscriber requires the notification).

> **Note**
>
> BRM stores policy attributes as subscriber preferences (**/profile/subscriber_preferences**) objects containing the following elements:
>
> - Name (required)
> - Type (optional)
> - Value (optional)

See "Configuring Subscriber Preferences" in *BRM ECE Implementing Charging* for information on storing subscriber preferences.

About the Default Provisioning Tag Provided by BRM

BRM provides a default provisioning tag in the **pin_offer_profile_provisioning_tags_policy_attributes.xml** file located in *BRM_home***/sys/data/config** directory.

The first section of the definition seen in Example 10-1 shows the valid resources for the default provisioning tag. It is followed by the information on the opcodes to be called when the product or discount containing the provisioning tag is purchased or canceled and the parameters that specify the fields to be added to the opcode's input flist and the value for each field.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Collecting Data for Provisioning Tags

Collect the following data for any additional services that BRM should track for policy-driven charging.

- The name for your provisioning tag. This must be the name of the offer profile with which you associate this provisioning configuration.
- Permitted types of services
- Subscriber preferences for the service type stored in the database as **/profile/subscriber_preferences** objects.

  For more information see "Configuring Subscriber Preferences" in *BRM ECE Implementing Charging*.
- Opcodes that are to be used in association with the provisioning tag

  You must set up the following for each of the opcodes you include in a provisioning tag:

  – The name and number of each opcode to run and when the opcode must be run. These opcodes contain the business logic to perform the actual provisioning, such as creating a profile.

  – Parameters that specify the fields to be added to each opcode's input flist and the value for each field.

This information can now be configured in the **pin_offer_profile_provisioning_tags_policy_attributes.xml** file for use with your offer profiles.

Configuring Provisioning Tags for Offer Profiles

To configure provisioning tags for use with your offer profiles:

1. Go to the *BRM_home***/sys/data/config** directory.
2. Open the **pin_offer_profile_provisioning_tags_policy_attributes.xml** file an XML editor or text editor.

   > **Tip**
   >
   > Save a copy of the default configuration file before you make any changes to it.

3. Do one of the following to update this file:
   - Edit the default provisioning tag contained in this file.
   - Add your provisioning tag by placing your definition just below the default provisioning tag (*Platinum*). Include the following for each provisioning tag:

     – Provisioning tag name

     ```
     <ProvisioningTag name="YourProvisioningTagName">
     ```

     where *YourProvisioningTagName* is the name of your offer profile.

     – Each service that is valid for the provisioning tag as a **PermittedType** element as shown in Example 10-1.

     – Specifications for the opcodes to call for this provisioning tag as shown in Example 10-1.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Tip**
>
> Table 10-3 lists the elements in the **pin_offer_profile_provisioning_tags_policy_attributes.xml** file, the syntax to use for each element, and a description of how to specify each element.

4. Save the **pin_offer_profile_provisioning_tags_policy_attributes.xml** file.

Modifying Provisioning Tags

You use provision tags as the names of offer profiles associated with products and discounts. If you modify a provision tag that is in use as an offer profile, be sure to modify the corresponding offer profile name accordingly.

See "Policy-Driven Charging" for information on offer profiles.

> **Note**
>
> - Do not remove existing provisioning tags from **pin_config_provisioning_tags.xml** when adding new tags unless you want existing tags removed from the **/config/provisioning_tag** objects in the database.
> - When you load **pin_config_provisioning_tags.xml**, all existing instances of **/config/provisioning_tag** are removed from the database, so only the provisioning tags defined in the file when you load it will be in the database.

Loading Provisioning Tag Configurations

If you are loading provisioning tags for policy-driven charging, see "Loading Provisioning Tags for Policy-Driven Charging".

After you configure provisioning tags in the **pin_config_provisioning_tags.xml** file, load the tags into the database with the **load_config_provisioning_tags** utility.

> **Note**
>
> The utility that loads provisioning tags into the database overwrites existing provisioning tags. When updating provisioning tags, you cannot load new provisioning tags only. You must load the complete set of provisioning tags each time you run the utility.

To load provisioning tag configurations:

1. Go to the directory in which the **pin_config_provisioning_tags.xml** file is located. The default location is *BRM_home***/sys/data/config**.
2. Use the following command to run the **load_config_provisioning_tags** utility:

```
load_config_provisioning_tags pin_config_provisioning_tags.xml
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> - When you run the utility, the**pin_config_provisioning_tags.xml**and **business_configuration.xsd** files must be in the same directory. By default, both files are in *BRM_home***/sys/data/config**.
> - This utility needs a configuration (**pin.conf**) file in the directory from which you run the utility.

If you do not run the utility from the directory in which **pin_config_provisioning_tags.xml** is located, include the complete path to the file, for example:

```
load_config_provisioning_tags BRM_home/sys/data/config/
pin_config_provisioning_tags.xml
```

For more information, see **load_config_provisioning_tags**.

3. Stop and restart the Connection Manager (CM).
4. To verify that the provisioning tags were loaded, display the **/config/provisioning_tag** object by using the Object Browser or the **robj** command with the **testnap** utility.

Loading Provisioning Tags for Policy-Driven Charging

Use the **load_config_provisioning_tags** utility to load the **pin_offer_profile_provisioning_tags_policy_attributes.xml** file into the BRM database.

To do so:

1. Ensure that the provisioning tags required for policy-driven charging are configured in the **pin_offer_profile_provisioning_tags_policy_attributes.xml** XML file.
2. Go to the directory in which the **pin_offer_profile_provisioning_tags_policy_attributes.xml** file is located. The default location is *BRM_home***/sys/data/config**.
3. Commit the provisioning tags and policy attributes XML file to the BRM database.

   ```
   load_config_provisioning_tags -d -v
   pin_offer_profile_provisioning_tags_policy_attributes.xml
   ```

   where:

   - **-d** creates a log file for debugging purposes.
   - **-v** displays information about successful or failed processing as the utility runs.

   For more information on the **load_config_provisioning_tags** utility, see "load_config_provisioning_tags" in *BRM Developer's Guide*.
4. To verify that the provisioning tags were loaded, display the **/config/provisioning_tag** object by using the Object Browser or the **robj** command with the **testnap** utility.
5. Stop and restart the Connection Manager (CM).

The provisioning tags and policy attributes information associated with policy-driven charging is now stored in **/config/provisioning_tag** objects in the database.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Modifying and Compiling the Provisioning Policy Source File

If a provisioning tag defined in the provisioning tag framework uses the PCM_OP_SUBSCRIPTION_POL_PURCHASE_PROD_PROVISIONING policy opcode, you must add the tag to the **fm_subscription_pol_provisioning.c** file and recompile the file.

A provisioning tag uses PCM_OP_SUBSCRIPTION_POL_PURCHASE_PROD_PROVISIONING opcode if it is included in a product and if it uses a service associated with the __DEFAULT__ provisioning tag. See "Sample Provisioning Tag XML File" for the list of permitted services for __DEFAULT__.

Modifying and compiling the provisioning policy source file enables PCM_OP_SUBSCRIPTION_POL_PURCHASE_PROD_PROVISIONING to handle the tags.

To modify the provisioning policy source file for provisioning tags created using the provisioning tag framework:

1. Open the *BRM_home***/source/sys/fm_subscription_pol/fm_subscription_pol_provisioning.c** file.
2. Add the provisioning tag name to the **service_info** table for the appropriate service.

   For example, to add a provisioning tag called *test* to **/service/ip**, change this code:

   ```
   static char *tags_ip[] = {
     "example",
     NULL    /* MUST BE LAST! */
   };
   ```

   to the following:

   ```
   static char *tags_ip[] = {
   "example",
   "test",
     NULL    /* MUST BE LAST! */
   };
   ```

3. Add code in the plp function to handle the new tag. You do this in the PROVISIONING FUNCTIONS section. The functions are grouped by service type.

   For example, to add the provisioning tag *test* to **/service/ip**, change this code:

   ```
   static void
   plp_ip(pcm_context_t *ctxp, poid_t *svc_obj_p, int32 buy,
     char *tag, pin_errbuf_t *ebufp)
   {
     if (strcmp(tag, "example") == 0) {
       plp_example(ctxp, svc_obj_p, buy, tag, ebufp);
     }
     else{
       plp_ssg(ctxp, svc_obj_p, buy, tag, ebufp);
     }
   }
   ```

   to the following:

   ```
   static void
   plp_ip(pcm_context_t *ctxp, poid_t *svc_obj_p, int32 buy,
     char *tag, pin_errbuf_t *ebufp)
   {
     if (strcmp(tag, "example") == 0) {
   ```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
      plp_example(ctxp, svc_obj_p, buy, tag, ebufp);
    }
    else if (strcmp(tag, "test") == 0) {
      /*skip*/
    }else{
      plp_ssg(ctxp, svc_obj_p, buy, tag, ebufp);
    }
  }
```

4. Compile and save the file.

Using a Policy Source File to Set Up Provisioning

You can define provisioning tags directly in the **fm_subscription_pol_provisioning.c** file without using the provisioning tag framework. This file is the single source file for all provisioning operations.

> **Note**
>
> The provisioning tags framework is the preferable method for creating provisioning tags.

To define provisioning tags directly in source code, follow these steps:

1. Open the *BRM_home***/source/sys/fm_subscription_pol/fm_subscription_pol_provisioning.c** file.
2. Define your provisioning tags by following the instructions in the file.
   a. In each entry in the **provisioning_tags** table, include the name of the service associated with the tag, the tag name, and calls to service-specific functions.
   b. Ensure that each function included in the table does the following:
      - Changes fields in the service object when customers purchase the service
      - Clears the appropriate fields when customers cancel the service
3. Compile and save the file.

If you create a provisioning tag in the policy source file, you must modify and recompile the source file to modify or delete the tag.

Sample Provisioning Tag XML File

Following is the default provisioning tag XML file. This file defines the provisioning tag named __DEFAULT__, which includes several permitted services and two opcodes:

```
<?xml version="1.0" encoding="UTF-8"?>

<BusinessConfiguration
    xmlns="http://www.portal.com/schemas/BusinessConfig"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.portal.com/schemas/BusinessConfig
business_configuration.xsd">

    <ProvisioningTagConfiguration>
    <ProvisioningTagList>

        <ProvisioningTag name="__DEFAULT__">
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
                <PermittedTypes>/service/email</PermittedTypes>
                <PermittedTypes>/service/ip/gprs</PermittedTypes>
                <PermittedTypes>/service/content</PermittedTypes>
                <PermittedTypes>/service/vpdn</PermittedTypes>
                <PermittedTypes>/service/ip</PermittedTypes>
                <PermittedTypes>/service/ip/gprs</PermittedTypes>
                <PermittedTypes>/service/telco</PermittedTypes>
                <PermittedTypes>/service/telco/gsm</PermittedTypes>
                <PermittedTypes>/service/telco/gsm/data</PermittedTypes>
                <PermittedTypes>/service/telco/gsm/fax</PermittedTypes>
                <PermittedTypes>/service/telco/gsm/sms</PermittedTypes>
                <PermittedTypes>/service/telco/gsm/telephony</PermittedTypes>

                <OpcodeList>
                    <OpcodeName>PCM_OP_SUBSCRIPTION_POL_PURCHASE_PROD_PROVISIONING</
OpcodeName>
                    <OpcodeNumber>417</OpcodeNumber>
                    <OpcodeMode>0</OpcodeMode>

                    <OpcodeParamsList>
                        <OpcodeParamName>PIN_FLD_POID</OpcodeParamName>
                        <OpcodeParamValue>$SERVICE$</OpcodeParamValue>
                    </OpcodeParamsList>
                    <OpcodeParamsList>
                        <OpcodeParamName>PIN_FLD_PROVISIONING_TAG</OpcodeParamName>
                        <OpcodeParamValue>$PROVTAG$</OpcodeParamValue>
                    </OpcodeParamsList>
                </OpcodeList>

                <OpcodeList>
                    <OpcodeName>PCM_OP_SUBSCRIPTION_POL_CANCEL_PROD_PROVISIONING</OpcodeName>
                    <OpcodeNumber>418</OpcodeNumber>
                    <OpcodeMode>1</OpcodeMode>

                    <OpcodeParamsList>
                        <OpcodeParamName>PIN_FLD_POID</OpcodeParamName>
                        <OpcodeParamValue>$SERVICE$</OpcodeParamValue>
                    </OpcodeParamsList>
                    <OpcodeParamsList>
                        <OpcodeParamName>PIN_FLD_PROVISIONING_TAG</OpcodeParamName>
                        <OpcodeParamValue>$PROVTAG$</OpcodeParamValue>
                    </OpcodeParamsList>
                </OpcodeList>

            </ProvisioningTag>

        </ProvisioningTagList>
        </ProvisioningTagConfiguration>

</BusinessConfiguration>
```

Default Provisioning Tag for Policy-Driven Charging

Example 10-1 shows the default provisioning tag provided by BRM in the **pin_offer_profile_provisioning_tags_policy_attributes.xml** file located in the *BRM_home*/**sys/data/config** directory.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Example 10-1 Default Provisioning Tag Configuration

```
<BusinessConfiguration xsi:schemaLocation="http://www.portal.com/schemas/BusinessConfig
business_configuration.xsd">
<ProvisioningTagConfiguration>
   <ProvisioningTagList>
     <ProvisioningTag name="Platinum">
        <PermittedTypes>/service/email</PermittedTypes>
        <PermittedTypes>/service/ip/gprs</PermittedTypes>
        <PermittedTypes>/service/content</PermittedTypes>
        <PermittedTypes>/service/vpdn</PermittedTypes>
        <PermittedTypes>/service/ip</PermittedTypes>
        <PermittedTypes>/service/fax</PermittedTypes>
        <PermittedTypes>/service/ip/gprs</PermittedTypes>
        <PermittedTypes>/service/telco</PermittedTypes>
        <PermittedTypes>/service/telco/gsm</PermittedTypes>
        <PermittedTypes>/service/telco/gsm/data</PermittedTypes>
        <PermittedTypes>/service/telco/gsm/fax</PermittedTypes>
        <PermittedTypes>/service/telco/gsm/sms</PermittedTypes>
        <PermittedTypes>/service/telco/gsm/sms</PermittedTypes>
        <OpcodeList>
            <OpcodeName>PCM_OP_CUST_SET_SUBSCRIBER_PREFERENCES</OpcodeName>
            <OpcodeNumber>3916</OpcodeNumber>
            <OpcodeMode>0</OpcodeMode>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_POID</OpcodeParamName>
                <OpcodeParamValue>0.0.0.0 /profile/subscriber_preferences -1</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_DELETED_FLAG</OpcodeParamName>
                <OpcodeParamValue>0</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_ACCOUNT_OBJ</OpcodeParamName>
                <OpcodeParamValue>$ACCOUNT$</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_SERVICE_OBJ</OpcodeParamName>
                <OpcodeParamValue>$SERVICE$</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_NAME</OpcodeParamName>
                <OpcodeParamValue>SET_SUBSCRIBER_PREFERENCES</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_SUBSCRIBER_PREFERENCES[0].PIN_FLD_NAME</OpcodeParamName>
                <OpcodeParamValue>Language</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_SUBSCRIBER_PREFERENCES[0].PIN_FLD_
VALUE</OpcodeParamName>
                <OpcodeParamValue>English</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_SUBSCRIBER_PREFERENCES[0].PIN_FLD_SUBSCRIBER_PREFERENCE_
ID</OpcodeParamName>
                <OpcodeParamValue>1</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_SUBSCRIBER_PREFERENCES[1].PIN_FLD_
NAME</OpcodeParamName>
                <OpcodeParamValue>Channel</OpcodeParamValue>
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
            </OpcodeParamsList>

            <OpcodeParamsList>
                 <OpcodeParamName>PIN_FLD_SUBSCRIBER_PREFERENCES[1].PIN_FLD_
VALUE</OpcodeParamName>
                 <OpcodeParamValue>IVR</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_SUBSCRIBER_PREFERENCES[1].PIN_FLD_SUBSCRIBER_PREFERENCE_
ID</OpcodeParamName>
                <OpcodeParamValue>2</OpcodeParamValue>
            </OpcodeParamsList>
        </OpcodeList>
        <OpcodeList>
            <OpcodeName>PCM_OP_CUST_SET_SUBSCRIBER_PREFERENCES</OpcodeName>
            <OpcodeNumber>3916</OpcodeNumber>
            <OpcodeMode>1</OpcodeMode>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_POID</OpcodeParamName>
                <OpcodeParamValue>0.0.0.0 /profile/subscriber_preferences -1</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_DELETED_FLAG</OpcodeParamName>
                <OpcodeParamValue>1</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_ACCOUNT_OBJ</OpcodeParamName>
                <OpcodeParamValue>$ACCOUNT$</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_SERVICE_OBJ</OpcodeParamName>
                <OpcodeParamValue>$SERVICE$</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_NAME</OpcodeParamName>
                <OpcodeParamValue>DEL_SUBSCRIBER_PREFERENCES</OpcodeParamValue>
            </OpcodeParamsList>
        </OpcodeList>
    </ProvisioningTag>
  </ProvisioningTagList>
</ProvisioningTagConfiguration>
</BusinessConfiguration>
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

11

Working with Extended Rating Attributes

This chapter describes how to use extended rating attributes (ERAs) in your Oracle Communications Billing and Revenue Management (BRM) system and describes how BRM rates events using ERAs.

Topics in this document:

- About Extended Rating Attributes
- Creating ERAs
- Creating Friends and Family ERAs
- Using ERAs with Multiple Lists
- About Rating Based on Friends and Family ERAs
- Customizing ERA Names and Descriptions for Client Applications

About Extended Rating Attributes

Extended rating attributes (ERAs) provide special rates or discounts based on a specific attribute of a service or account, such as a telephone number. For example, you use ERAs to offer special friends and family rates or a birthday discount.

You can create two types of ERAs:

- **Service-level**: An ERA that is used with a specific service, such as a telco service. Special rates for friends and family and closed user group telephone calls are examples of service-level ERAs.

  > **Note**
  >
  > A service-level ERA can be assigned to a service that represents a subscription so that it applies to all services associated with a customer's subscription.

- **Account-level**: An ERA that can be used with any service (for example, a birthday discount).

You create an ERA by adding it to a provisioning tag.

ERAs must be configured with specific data for each customer. For example, to set up a birthday discount, you must specify the customer's birthday.

You configure ERAs in Customer Center, where ERAs are called *promotions*. ERAs you configure are stored as profiles.

For more information about ERAs, see:

- About Multiple ERA Lists
- About Configuring ERAs in Customer Center
- About Sharing ERAs

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Creating ERAs

About Multiple ERA Lists

For some types of ERAs, such as friends and family, you can create multiple lists for the same ERA. Each list is identified as a *label* in the BRM system. This enables you to set up rating and discounting to select the rate or discount based on the label name.

For example, you can design a friends and family ERA for a GSM telephony service that charges different rates for friends and family using the labels in Table 11-1:

Table 11-1 Example Labels and Rates

ERA Label	Rate
MYFAMILY	1 cent per minute
MYFRIENDS	2 cents per minute

You include labels when you define the ERA in a provisioning tag.

You use price selectors, discount selectors, or charge selectors to specify different rates or discounts for each label.

About Configuring ERAs in Customer Center

Use Customer Center to configure an ERA for an individual customer. When a customer purchases a charge offer or discount that includes a provisioning tag with an ERA, that ERA is displayed in Customer Center as a *promotion*. You specify values for the ERA on the **Promotion** tab.

When you configure an ERA in Customer Center, the values are stored in one of these profile objects:

- **/profile/serv_extrating** for a service-level ERA
- **/profile/acct_extrating** for an account-level ERA

> **Note**
>
> – Even though an account is qualified to use ERAs, you do not have to implement them in the account.
>
> – BRM does not validate any data entered when configuring ERAs (for example, telephone numbers for the friends and family discount). To create validation rules for these entries, edit the PCM_OP_CUST_POL_PREP_PROFILE policy opcode.
>
> – ERA codes are defined in Pipeline Manager configuration files. The types of values you enter in Customer Center depends on how you configure the ERAs in Pipeline Manager. For example, if the ERA is configured to match a telephone number in an event, you would enter telephone numbers in Customer Center.
>
> – When configuring ERAs in Customer Center, use only uppercase letters, ASCII 7-bit punctuation, and no spaces.

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Sharing ERAs

You can share the ERA values you configure for one service with other services by using profile sharing groups. For example, you can share the same list of phone numbers for a friends and family ERA among all the customers whose phone numbers are included in the list.

You can share service-level ERAs only.

Creating ERAs

This section is an overview of the main steps you take to create an ERA.

Each of the following steps is a separate task. Use the links for detailed information about each task:

1. **Create a provisioning tag.** ERAs are defined in provisioning tags, which you can create in a few different ways.
2. **Define how the ERA is rated.** For example, an ERA can be rated based on the usage type or the ERA label name.
3. **Add the ERA's provisioning tag to a charge offer or discount offer**.
4. **Add the name and description to the ERA description file.** The name and description appears in Customer Center. For provisioning tags created with the provisioning tag framework, you must take additional steps to set up the name and description. See "Customizing ERA Names and Descriptions for Client Applications".
5. **Configure the ERA for an individual customer.** See "About Configuring ERAs in Customer Center".
6. **(Optional) Create a profile sharing group to share the ERA configuration with other accounts or services.** See "About Sharing ERAs ".

> **Note**
>
> You can share service-level ERAs only.

Creating Friends and Family ERAs

BRM has built-in support for friends and family ERAs. This feature enables you to create an ERA with multiple lists, each containing phone numbers, APN addresses, email addresses, or other values. You can also use profile sharing to share a friends and family ERA with multiple services.

Following is a description of the steps for creating a friends and family ERA. Use the links for more detailed information about carrying out each task:

1. Create a provisioning tag.

   Create a provisioning tag with a service-level ERA called FRIENDS_FAMILY and one or more labels, which are ERA lists. The labels are optional, but they give you the flexibility to define multiple lists for an ERA and to use the label name as a rule for model selectors or charge selectors.

   You create the provisioning tag in different ways depending on its service:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- For any service, you can use the provisioning tag framework. This involves defining the tag in the **pin_config_provisioning_tags.xml** file.

  The definition of the tag includes one or more opcodes that are run when the provisioning tag is purchased. For ERAs, you can use PCM_OP_SUBSCRIPTION_PROVISION_ERA. This opcode creates a **/profile** object for the ERA.

- For telco services, use the Provisioning Tags application, or modify and load the service-specific version of the **pin_telco_tags** file.

> **Note**
>
> For telco services, you cannot use the provisioning tag framework to include supplementary services and service extensions in the same provisioning tag with the ERA.

2. Enable **pin_notify** entries.

   a. Uncomment the following entries in the **pin_notify** file:

   ```
   3788   0     /event/group/sharing/profiles/create
   3788   0     /event/group/sharing/profiles/modify
   ```

   b. Load the **pin_notify** file and restart the Connection Manager (CM).

3. For pipeline rating, configure the ISC_ProfileAnalyzer iScript if needed.

   ISC_ProfileAnalyzer matches a value in an EDR, such as a phone number, with lists in the friends and family ERA. It then adds the names of ERA labels that include that value to the EDR.

   By default, this iScript analyzes an ERA named FRIENDS_FAMILY that uses a GSM telephony service. You can customize the iScript to handle a different service or ERA name.

   To enable ISC_ProfileAnalyzer, configure the FCT_iScript module and set the **Active** entry to **True**.

> **Note**
>
> For rerating, ISC_ProfileAnalyzer must be enabled. The default registry file is *BRM_home***/conf/wirelessRealtime.reg**.

   See "Customizing Pipeline Rating for Friends and Family ERAs".

4. For pipeline rating, create a price selector or discount selector.

   For pipeline events, use a price or discount selector to specify different rates or discounts for different ERA labels. See "Using ERAs with Multiple Lists".

5. Create a charge offer or discount offer that includes the provisioning tag.

   For a charge offer, include the charge with the price selector or charge selector you defined. For a discount, include the discount selector you defined.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> - You cannot use provisioning tags created with the Services Framework Manager with discounts.
> - Use the provisioning tag framework to create an ERA that you want to add directly to a discount.

6. Add the names and descriptions to the ERA description file.

   Names and descriptions from this file are displayed in Customer Center and Provisioning Tags. See "Customizing ERA Names and Descriptions for Client Applications".

7. Enter values for the friends and family promotion for a specific customer who purchases a charge offer or discount offer with the friends and family provisioning tag.

> **Note**
>
> If you do not add name-value pairs for friends and family ERAs, DAT_AccountBatch does not load the friends and family ERAs into the pipeline memory at startup.

   Promotion is the term used for *ERA* in Customer Center. For example, for a telephony service, you would configure the promotion by entering phone numbers. For other services, you would enter the values relevant for the service, such as APN or email addresses.

   If the ERA contains multiple lists, you configure each list in Customer Center.

8. To share ERA lists between accounts, create a profile sharing group.

9. (Optional) Create a profile sharing group to share the friends and family lists with other accounts or services.

> **Note**
>
> You can share service-level ERAs only.

   When you configure a promotion in Customer Center, the values are stored in a profile. You can share that profile with services owned by accounts that own a charge offer or discount offer with the same ERA.

> **Note**
>
> Typically, you would set up rating or discounting to use either the ERA label name with a model selector or the usage type. If you are using ERA labels and model selectors, you should comment out the IRL_UsageType iRule from the registry.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Sample Friends and Family Provisioning Tag

This is sample XML code for a provisioning tag that creates a friends and family ERA for an email service.

This code will be added to the **pin_config_provisioning_tags.xml** file:

```
<ProvisioningTag name="FRIENDS_FAMILY">

            <PermittedTypes>/service/email</PermittedTypes>

            <OpcodeList>
                <OpcodeName>PCM_OP_SUBSCRIPTION_PROVISION_ERA</OpcodeName>
                <OpcodeNumber>9066</OpcodeNumber>
                <OpcodeMode>0</OpcodeMode>

                <OpcodeParamsList>
                    <OpcodeParamName>PIN_FLD_POID</OpcodeParamName>
                    <OpcodeParamValue>0.0.0.0 /profile/serv_extrating -1</
OpcodeParamValue>
                </OpcodeParamsList>

                <OpcodeParamsList>
                    <OpcodeParamName>PIN_FLD_ACCOUNT_OBJ</OpcodeParamName>
                    <OpcodeParamValue>$ACCOUNT$</OpcodeParamValue>
                </OpcodeParamsList>

                <OpcodeParamsList>
                    <OpcodeParamName>PIN_FLD_FLAGS</OpcodeParamName>
                    <OpcodeParamValue>0</OpcodeParamValue>
                </OpcodeParamsList>

                <OpcodeParamsList>
                    <OpcodeParamName>PIN_FLD_SERVICE_OBJ</OpcodeParamName>
                    <OpcodeParamValue>$SERVICE$</OpcodeParamValue>
                </OpcodeParamsList>

                <OpcodeParamsList>
                    <OpcodeParamName>PIN_FLD_NAME</OpcodeParamName>
                    <OpcodeParamValue>FRIENDS_FAMILY</OpcodeParamValue>
                </OpcodeParamsList>

                <OpcodeParamsList>

<OpcodeParamName>PIN_FLD_INHERITED_INFO.PIN_FLD_EXTRATING.PIN_FLD_LABEL</OpcodeParamName>
                    <OpcodeParamValue>MYFRIENDS</OpcodeParamValue>
                </OpcodeParamsList>
            </OpcodeList>

            <OpcodeList>
                <OpcodeName>PCM_OP_SUBSCRIPTION_PROVISION_ERA</OpcodeName>
                <OpcodeNumber>9066</OpcodeNumber>
                <OpcodeMode>1</OpcodeMode>

                <OpcodeParamsList>
                    <OpcodeParamName>PIN_FLD_POID</OpcodeParamName>
                    <OpcodeParamValue>0.0.0.0 /profile/serv_extrating -1</
OpcodeParamValue>
                </OpcodeParamsList>

                <OpcodeParamsList>
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
                        <OpcodeParamName>PIN_FLD_ACCOUNT_OBJ</OpcodeParamName>
                        <OpcodeParamValue>$ACCOUNT$</OpcodeParamValue>
                    </OpcodeParamsList>

                    <OpcodeParamsList>
                        <OpcodeParamName>PIN_FLD_FLAGS</OpcodeParamName>
                        <OpcodeParamValue>1</OpcodeParamValue>
                    </OpcodeParamsList>

                    <OpcodeParamsList>
                        <OpcodeParamName>PIN_FLD_SERVICE_OBJ</OpcodeParamName>
                        <OpcodeParamValue>$SERVICE$</OpcodeParamValue>
                    </OpcodeParamsList>

                    <OpcodeParamsList>
                        <OpcodeParamName>PIN_FLD_NAME</OpcodeParamName>
                        <OpcodeParamValue>FRIENDS_FAMILY</OpcodeParamValue>
                    </OpcodeParamsList>

                    <OpcodeParamsList>

<OpcodeParamName>PIN_FLD_INHERITED_INFO.PIN_FLD_EXTRATING.PIN_FLD_LABEL</OpcodeParamName>
                        <OpcodeParamValue>MYFRIENDS</OpcodeParamValue>
                    </OpcodeParamsList>
                </OpcodeList>

</ProvisioningTag>
```

> **Note**
>
> The PCM_OP_SUBSCRIPTION_PROVISION_ERA opcode is specified twice, once with OpcodeMode and PIN_FLD_FLAGS set to **0** to run at purchase time and once with OpcodeMode and PIN_FLD_FLAGS set to **1** to run at cancel time. This opcode requires using either **0** or **1** for these values.

To create an ERA with a second list, add two more <OpcodeList> blocks using the same opcode and the same values, except for a different value for PIN_FLD_INHERITED_INFO.PIN_FLD_EXTRATING.PIN_FLD_LABEL. Do the same if you want additional lists for the ERA.

Using ERAs with Multiple Lists

You can have multiple rates or discounts for an ERA with multiple ERA labels. An ERA label is an individual list of values, such as phone numbers or email addresses. A single ERA can have multiple ERA labels.

For example, a friends and family ERA can have separate lists for friends and family, each with a different rate. During rating or discounting, BRM chooses the correct rate based on the label name.

Using Model Selectors for ERAs

You can use price or discount selectors to define different rates or discounts for each ERA label in an ERA. Create model selector rules based on the PROFILE_LABEL_LIST EDR field. For each ERA label name, you can specify a different price or discount. During rating or

Copyright © 2025, 2026, Oracle and/or its affiliates.

discounting, Pipeline Manager matches a value in the EDR such as a phone number with the values in the ERA labels.

Model selectors also rank the price or discounts. If the EDR matches more than one ERA label, the first pricing whose rule matches a label in the EDR is used.

In the following example, a friends and family ERA has two labels, MYFAMILY and MYFRIENDS. Each label is associated with a different pricing through a price selector, as shown in Table 11-2:

Table 11-2 Example of Friends and Family ERA

Rank	DETAIL.PROFILE_LABEL_LIST	Pricing
1	*MYFAMILY*	1_cent_per_min
2	*MYFRIENDS*	2_cent_per_min

> **Note**
>
> To use a discount with a discount selector for a telco service, use the provisioning tag framework to create the tag with the ERA. You cannot use provisioning tags created with the Services Framework Manager with discounts.

Using Charge Selectors for ERAs

You can define different rates for each ERA label in an ERA for real-time rating by using charge selectors. To do this, use the event attribute EVENT.PIN_FLD_PROFILE_LABEL_LIST in the charge selector matrix.

In the following example, a friends and family ERA has two labels, MYFAMILY and MYFRIENDS. Each label is associated with a different rate through a charge selector, as shown in Table 11-3:

Table 11-3 Charges for ERAs

Row	EVENT.PIN_FLD_PROFILE_LABEL_LIST	Charge	Impact Category
1	*MYFAMILY*	1_cent_per_min	Default
2	*MYFRIENDS*	2_cent_per_min	Default

About Rating Based on Friends and Family ERAs

This section describes how BRM rates events with friends and family ERAs. For general information about ERAs, see "About Extended Rating Attributes". For information about working with multiple ERA lists, see "Using ERAs with Multiple Lists".

In both real-time rating and pipeline rating, the following occurs:

- If a predefined value in an event or EDR matches a value in an ERA, the names of the ERA and ERA labels are added to the event or EDR.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- If a charge selector or model selector has been defined based on the ERA label name, that selector is used to determine the rate. If an event has multiple labels, the charge selector or model selector also determines which label has priority.
- By default, friends and family ERAs are rated for telco services, but through customization they can be used and rated for any service.

For more information, see:

- Real-Time Rating for Friends and Family ERAs
- Pipeline Rating for Friends and Family ERAs

Real-Time Rating for Friends and Family ERAs

Friends and family ERAs can be rated based on either the ERA name, which is specified as the usage type, or the ERA label name.

In real-time rating, if a particular event attribute matches a value in a friends and family ERA, the ERA is added to the event's usage type field, and the ERA label is added to the event's ERA label field. An event can match more than one friends and family label.

Using the ERA label name enables you to use different rates for different values within the same ERA. For example, you can use labels to charge different rates for calls to friends and calls to families by defining a charge selector. Using the usage type, you can charge a single rate for all friends and family calls.

Following is a summary of the real-time rating process for events with friend and family ERAs. This section contains information specific to friends and family ERAs.

1. The PCM_OP_RATE_POL_PRE_RATING policy opcode calls the PCM_OP_RATE_POL_PROCESS_ERAS policy opcode, which in turn calls PCM_OP_RATE_GET_ERAS.
2. PCM_OP_RATE_GET_ERAS retrieves the valid ERAs for an event, including ERAs for the event's account, service, and shared profiles. The retrieved information includes the ERA name, label names, label values, and validity dates.
3. The PCM_OP_RATE_POL_PROCESS_ERAS policy opcode compares certain predefined event fields to the values for the ERA or ERAs valid for the event.

   By default, PCM_OP_RATE_POL_PROCESS_ERAS is configured to work with a GSM service and to compare values in the PIN_FLD_CALLING_FROM and PIN_FLD_CALLED_TO fields.

   PCM_OP_RATE_POL_PROCESS_ERAS can be easily configured to work with a GPRS service. In that case, it compares values in the PIN_FLD_ANI and PIN_FLD_DN fields.

   > **Note**
   >
   > Using the PCM_OP_RATE_POL_PROCESS_ERAS policy opcode with services other than GSM and GPRS requires more extensive customization.

4. If a number in the PIN_FLD_CALLED_TO field matches a value in one or more of the friends and family ERA labels, PCM_OP_RATE_POL_PROCESS_ERAS adds the following to the event:
   - The ERA name to the PIN_FLD_USAGE_TYPE field. For a friends and family ERA, the name FF is added.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- The ERA label name or names to the PIN_FLD_PROFILE_LABEL_LIST field. The names are added as they were defined in the ERA (for example, MYFRIENDS and MYFAMILY). Multiple names are separated by a comma, unless you specified a different separator by customizing the opcode.

The charge or discount for the event is then calculated based on the value of either the PIN_FLD_USAGE_TYPE or PIN_FLD_PROFILE_LABEL_LIST field.

If a charge selector was used, the rating engine uses it to find the correct rate. For a friends and family ERA, typically the charge selector would use the PIN_FLD_PROFILE_LABEL_LIST event field to map to different rates. For example, you could have a different rate for each ERA label.

Customizing Real-Time Rating for Friends and Family ERAs

You can customize the PCM_OP_RATE_POL_PROCESS_ERAS policy opcode to do the following:

- Add a service type, so you can rate ERAs for services other than telco services.

  To do this, modify the **fm_rate_pol_proc_eras_set_era_name()** function.

  To add a new service type, copy the existing code for SERVICE_TELEPHONY. To add a new service type, copy the code. In the copied code, change the service type, and change the PIN_FLD_CALLING_FROM and PIN_FLD_CALLED_TO fields to the appropriate fields for the new service type.

  This is an excerpt from the default version of **fm_rate_pol_process_eras.c** showing the code you must copy and modify:

```
if(svc && !strncmp(svc,SERVICE_TELEPHONY,strlen(SERVICE_TELEPHONY))) {
  column_flistp = PIN_FLIST_ELEM_GET(e_flistp,PIN_FLD_CALL, rec_id, 1, ebufp);
  if(column_flistp) {
    a_num = PIN_FLIST_FLD_GET( column_flistp, PIN_FLD_ANI, 1, ebufp );
    b_num = PIN_FLIST_FLD_GET( column_flistp, PIN_FLD_DNIS, 1, ebufp );
}
```

- Change the separator character used to separate multiple label names in the PIN_FLD_PROFILE_LABEL_LIST field. Comma is the default. You specify this character in the CM_OP_RATE_POL_PROCESS_ERAS policy opcode and in the charge selector. Specify the same character in both places.

Prioritizing Rates for ERAs in Real-Time Rating

The charge selector also ranks the charges. If the event matches more than one ERA label, the rating engine evaluates the rates in order of their rank in the charge selector and chooses the first rate that matches a label in the EDR.

The charge selector matches ERA labels within a list of multiple labels by using the *In List* matching type. This must be selected when you create the charge selector. The rating engine then treats PIN_FLD_PROFILE_LABEL_LIST or whichever fields you include in the charge selector as a list and parses it accordingly, using a separator you specify.

In the following example, the friends and family ERA has two labels, MYFAMILY and MYFRIENDS. Each label is associated with a different rate through a charge selector, as shown in Table 11-4:

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 11-4 Charges and Multiple Labels

Row	EVENT.PIN_FLD_PROFILE_LABEL_LIST	Charge	Impact Category
1	*MYFAMILY*	1_cent_per_min	Default
2	*MYFRIENDS*	2_cent_per_min	Default

If an event's PIN_FLD_PROFILE_LABEL_LIST is **MYFAMILY**, the rating engine chooses the **1_cent_per_min** charge. If PIN_FLD_PROFILE_LABEL_LIST contains the value **MYFAMILY, MYFRIENDS**, the rating engine again chooses the **1_cent_per_min** charge. Both charges are matched, but **1_cent_per_min** has higher priority.

Pipeline Rating for Friends and Family ERAs

In pipeline rating, if ERA labels are used and a particular EDR field matches a value in a friends and family list, the ISC_ProfileAnalyzer iScript adds the ERA label to the EDR. An EDR can match more than one friends and family label. You set up a price or discount selector to select a price or discount based on the ERA label name. See "Using Model Selectors for ERAs".

If the IRL_UsageType iRule has been configured for ERAs, and a usage type set up in Pricing Center or PCC, the ERA name (not the label name) is added to the EDR's usage type field.

Typically, you would set up rating or discounting to use either the ERA label name with a model selector or the usage type. If you are using ERA labels and model selectors, you should comment out the IRL_UsageType iRule from the registry.

Following is a summary of the pipeline rating and discounting process for events with friend and family ERAs:

1. At startup, DAT_AccountBatch gets ERA and shared profile information from the BRM database and stores it in memory.
2. When a CDR is rated, DAT_AccountBatch determines the ERAs or profile sharing group that applies to the CDR, filters out the profiles specified in the **ProfilesNotLoadedIntoEdr** registry entry, and sends the remaining profile information to the FCT_Account module.
3. FCT_Account adds ERA and shared profile data to the EDR.
4. The ISC_ProfileLabel iScript calls the search method in DAT_AccountBatch to determine if any of the profiles that were filtered contains a value that matches the EDR field value. See "Improving Pipeline Rating Performance for Events with ERAs".

   If there's a match, ISC_ProfileLabel does the following:

   a. Parses the ERA labels returned by the search method and populates them into the DETAIL.PROFILE_LABEL_LIST EDR field.

   b. Populates the DETAIL.USAGE_TYPE EDR field based on the matching ERAs found.
5. For the profiles not specified in the ProfilesNotLoadedIntoEdr registry entry, the following occurs:

   a. ISC_ProfileAnalyzer iscript analyzes the profiles and determines if any of the profiles contains a value that matches the EDR field value. If there is a match, the ERA label is added to the DETAIL.PROFILE_LABEL_LIST field in the EDR container.

      By default, ISC_ProfileAnalyzer analyzes ERA profiles named FRIENDS_FAMILY for the service codes TEL (for GSM telephony) or SMS, by comparing the value of the

Copyright © 2025, 2026, Oracle and/or its affiliates.

DETAIL.B_NUMBER (called number) EDR field to the values in the ERA. You can customize ISC_ProfileAnalyzer to analyze other ERA profiles and service types. See "Customizing Pipeline Rating for Friends and Family ERAs".

6. The IRL_UsageType iRule adds the usage type to the DETAIL.USAGE_TYPE field in the EDR container if the conditions in the IRL_UsageType.irl file are met. For a friends and family ERA, the name added to USAGETYPE is FF.
7. For rating, FCT_MainRating finds the correct pricing for calculating the charge as follows:
   - If there is an ERA name in the USAGETYPE field, FCT_MainRating uses that ERA to rate the event.
   - If one or more ERA labels are listed in PROFILE_LABEL_LIST, FCT_MainRating chooses pricing using the rules in the price selector.
8. For discounting, FCT_DiscountAnalysis finds the correct discount to calculate the discount, using the values in the PROFILE_LABEL_LIST EDR field and the rules in the discount selector.

   The FCT_Discount module applies the discount after FCT_DiscountAnalysis determines the discount to use.

Prioritizing Rates and Discounts for ERAs in Pipeline Rating

The price or discount selector ranks the price or discounts. If the EDR matches more than one ERA label, Pipeline Manager evaluates the price or discounts in order of ranking and chooses the first price or discount whose rule matches a label in the EDR.

In the following example, the friends and family ERA has two labels, MYFAMILY and MYFRIENDS. Each label is associated with a different pricing through a price selector, as shown in Table 11-5:

Table 11-5 Price Selector

Rank	DETAIL.PROFILE_LABEL_LIST	Pricing
1	*MYFAMILY*	1_cent_per_min
2	*MYFRIENDS*	2_cent_per_min

If an EDR's DETAIL.PROFILE_LABEL_LIST is **MYFAMILY**, the rating module chooses the **1_cent_per_min** pricing. If DETAIL.PROFILE_LABEL_LIST contains the value **MYFAMILY, MYFRIENDS**, FCT_MainRating again chooses the **1_cent_per_min** pricing. Both pricings are matched, but **1_cent_per_min** has higher priority.

Improving Pipeline Rating Performance for Events with ERAs

When Pipeline Manager processes a call detail record (CDR), all the ERA profiles (owned or shared) that apply to the CDR are loaded into the event data record (EDR) container. However, some accounts may own a large number of ERA profiles or share a large number of ERA profiles in a profile sharing group. The populating of a large number of profiles into the EDR container may result in performance degradation. For example, when a Friends and Family profile sharing group contains a large number of ERA profiles that are shared with accounts in a hierarchy, all the shared profiles in the group are populated into the EDR container during CDR processing of the member accounts. In this example, the population of all the shared profiles into the EDR containers for *each* account in the hierarchy results in performance degradation.

Copyright © 2025, 2026, Oracle and/or its affiliates.

When you have accounts that own or share a large number of ERA profiles, you can use the DAT_AccountBatch registry entry, **ProfilesNotLoadedIntoEdr**, to specify the profiles to be filtered. When a CDR is rated, if the value in the EDR field matches a value in one or more ERA profiles in the **ProfilesNotLoadedIntoEdr** registry entry, the ISC_ProfileLabel iScript loads only the ERA labels into the EDR container.

To configure batch pipeline rating to filter ERA profiles:

1. Set the DAT_AccountBatch module registry entry **ProfilesNotLoadedIntoEdr** to the profiles that should be filtered. You can specify any number of profiles separated by a comma.

   For example:

   ```
   ProfilesNotLoadedIntoEdr {FRIENDS_FAMILY, DISCOUNTBUNDLE}
   ```

2. Configure the ISC_ProfileLabel iScript.

Customizing Pipeline Rating for Friends and Family ERAs

You customize pipeline rating for friends and family ERAs by customizing the ISC_ProfileAnalyzer and ISC_ProfileLabel iScripts.

By default, ISC_ProfileAnalyzer and ISC_ProfileLabel analyzes ERAs named FRIENDS_FAMILY for the service codes TEL (for GSM telephony) or SMS, comparing the value of the DETAIL.B_NUMBER (called number) field in the EDR to the ERA values.

To use ISC_ProfileAnalyzer and ISC_ProfileLabel with other services and ERAs, add an ELSE IF block to the onDetailEdr function. The default version of ISC_ProfileAnalyzer includes this example for adding a GPRS service:

```
else
if (edrServiceType == "GPRS")
{
   compareString = edrString(DETAIL.ASS_GPRS_EXT.APN_ADDRESS,0);
   compProfName  = "FRIENDS_FAMILY";
}
```

This block includes the entries shown in Table 11-6:

Table 11-6 Entries for GPRS Service

String	Description
edrServiceType	The service code.
compareString	The EDR field to compare to the ERA.
compProfName	The name of the ERA to analyze to compare to the EDR field.

Then you edit this information in the registry:

- Service type: Change TEL to the new service code.
- Profile name: Change FRIENDS_FAMILY to a different ERA name if needed.
- Separator: Change a comma to another character to separate multiple label names in EDRs, if needed.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Customizing ERA Names and Descriptions for Client Applications

You customize ERA names and descriptions that are displayed on the **Promotion** tab in Customer Center by editing the **era_descr.en_US** file in the *BRM_home*/**sys/msgs/eradescr** directory. You can localize this file.

> **Note**
>
> For ERAs defined in telco provisioning tags, the names and descriptions are also used in the Provisioning Tags application.

This file includes entries for ERA names and descriptions that are identified by ID numbers. Each name and description must have a unique ID number.

You define the ID numbers differently depending on how the ERAs are defined:

- For ERAs defined in telco provisioning tags, the ID numbers must match the ID numbers used in **pin_telco_tags_gsm** or other **pin_telco_tags** files that you edit when configuring ERAs.

  For example, this is the entry for the SPECIAL_DAY account-level ERA in the **pin_telco_tags_gsm** file:

  ```
  account_era  "SPECIAL_DAY"  2  3
  ```

- For ERAs defined using the provisioning tag framework, you include each ID in the **customized.properties** file in the Customer Center SDK. You also must include the name of the ERA in the provisioning tag definition.

  See "Creating Names and Descriptions for ERAs Defined in the Provisioning Tag Framework".

The following example shows the name and description of the Special Day ERA in the **era_descr.en_US** file:

```
ID = 2 ;
VERSION = 1 ;
STRING = "Special Day" ;

ID = 3 ;
VERSION = 1 ;
STRING = "This option gives discounts on calls made on your birthday. An account can
have only one Birthday. To give this promotion, enter the word BIRTHDAY in the Name
field, and enter the customer's birthday in the Value field." ;
```

When you add or change ERA names and descriptions, you might also need to provide more detailed information for your CSRs. For example, you could provide a list of valid entries for the Customer Type ERA and instructions on when to use each entry.

After you customize the file, use the **load_localized_strings** utility to load the contents of the **era_descr.en_US** file, or a localized version of the file, into the **/strings** object.

To load the contents of **era_descr.en_US**, use this command:

```
load_localized_strings era_descr.en_US
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> - ERA label names are not included in the **era_descr.en_US** file and cannot be localized.
> - Default ERA names and descriptions are loaded when you install a service manager. You must load them again only if you customize them.
> - If you're loading a localized version of this file, use the correct file extension for your locale.

Creating Names and Descriptions for ERAs Defined in the Provisioning Tag Framework

If you define an ERA using the provisioning tag framework, use the following procedure to create a name and description that appears in Customer Center:

1. Add the name and description to the **era_descr.en_US** file and to any other localized versions of the ERA description file you are using.

   For information, see "Customizing ERA Names and Descriptions for Client Applications".

   > **Note**
   >
   > - You must use a unique ID number for each name and description you add.
   > - You must use the same ID number for the same ERA in each localized version of the ERA description file.

2. Load the names and descriptions using the **load_localized_strings** utility.

   See "Customizing ERA Names and Descriptions for Client Applications".

3. Configure the ERA in the provisioning tag framework as follows:
   - Configure the provisioning tag to call the PCM_OP_SUBSCRIPTION_PROVISION_ERA opcode during purchase and cancel modes.
   - Specify the ERA name in the PIN_FLD_NAME parameter for the opcode. For example, for a friends and family ERA, specify **FRIENDS_FAMILY** in the provisioning tags configuration file.
   - Set the PIN_FLD_STR_VAL parameter to **12, 13** so that the profile name and profile description are localized and are stored in the **/string** object.

Copyright © 2025, 2026, Oracle and/or its affiliates.

12

Working with Price Tags

This chapter describes how to use price tags in your Oracle Communications Billing and Revenue Management (BRM) system and describes how BRM uses price tags to apply special prices at purchase time.

Topics in this document:

- About Price Tags
- Creating Price Tag Configuration Objects
- Modifying Price Tag Configuration Objects
- Using Price Tags With Sequential Discounts

About Price Tags

Price tags are components that let you apply special prices to products and discounts according to your specified rules.

You configure price tags in configuration objects, add them to balance impacts when creating products and discounts, and set their values at purchase time.

The rules configured in the configuration objects determine which price tags are valid for a given balance impact and which values can be used at purchase time. You cannot use price tags with chargeshares.

When you configure price tag configuration objects, you define the following:

- The price tag name.
- The price tag description.
- What type of values are allowed. Can be any value, a list of values, or a range of values.
- The values that are allowed, according to the specified type.
- The resources you add the price tag to. For example, Free Minutes, US Dollars, or any resource.
- The unit of measurement you can use the price tag with. For example, gigabytes, hours, amount used, or any unit.
- The services you can use the price tag with. For example, TelcoGSM, Email, or any service.

You create the configuration objects **config_price_tags.xml** files and load them using the **load_config** utility. BRM stores price tag configuration information in the **/config/price_tags** storable class. See "Creating Price Tag Configuration Objects".

After configuring price tag configuration objects, you can update or delete them under certain conditions. See "Modifying Price Tag Configuration Objects".

You add price tags to products and discounts in your price list and BRM validates them against the configuration objects. You can add the price tags using:

- The XML pricing interface. See "Adding Price Tags to Products and Discounts".

Copyright © 2025, 2026, Oracle and/or its affiliates.

- A third-party client that calls the PCM_OP_PRICE_SET_PRICE_LIST opcode. Add the price tags by setting the PIN_FLD_PRICE_TAG field under the balance impacts to the name of the appropriate price tag. See "Committing Price List Data to the BRM Database" for more information about PCM_OP_PRICE_SET_PRICE_LIST.

You can use the PCM_OP_SUBSCRIPTION_GET_PRICE_TAGS opcode to get the list of price tags that are valid for a resource, unit of measure, service, or product. See "Getting Information about Price Tags" in *BRM Opcode Guide* for more information about this opcode.

At purchase time, you specify values for the price tags. BRM validates the values entered against those configured in the configuration objects and stores the price tag details, including associated events, date ranges, and amounts, in **/offering_override_values** objects. See "Overriding Charges and Discounts for a Period of Time" in *BRM Opcode Guide* for more information about this opcode for more information about how these objects are updated.

Creating Price Tag Configuration Objects

You create price tag configuration objects in a **config_price_tags.xml** files and load them using the **load_config** utility.

To create price tag configuration objects:

1. Open the *BRM_home***/sys/data/config/config_price_tags.xml** file in a text editor.
   The file contains sample price tags, but you can modify them to meet your business needs.
2. Add a **PRICE_TAGS** array element for every new tag you want to create.

   Table 12-1 describes the elements in the price tag configuration file.

Table 12-1 Elements in the PRICE_TAGS element

Element	Description
<NAME>	The name of the price tag.
<DESCR>	The description of the price tag.
<RULE_TYPE>	The type of values for the price tag: • **ANY**: Lets you enter any value as a string. This is the default. • **LIST**: Lets you specify a list of values. • **RANGE**: Lets you specify a range of values.
<CONSTRAINTS>	The valid values, in the format that corresponds to the rule: • If **<RULE_TYPE>** is set to **ANY**, this element is not required, and any value can be entered at purchase time. • If **<RULE_TYPE>** is set to **LIST**, enter a semicolon separated list of values. For example, **10;20;30**. The default value is **0**, which allows all values. • If **<RULE_TYPE>** is set to **RANGE**, enter the values for the bottom and top of the range, separated by a semicolon. For example, **10:30**. The default value is **0:0**, which allows all values. Values can be positive or negative integers or decimals.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 12-1 (Cont.) Elements in the PRICE_TAGS element

Element	Description
<RESOURCE_ID>	The resource ID for the balance elements you can use the price tag with. For example, **840** for US Dollars, or **0** to add the price tag to multiple balance elements.
<RESOURCE_UNIT>	The resource unit for the balance elements that you can use the price tag with. Valid units are: • BYTE • KILOBYTE • MEGABYTE • GIGABYTE • DAY • HOUR • MINUTE • SECOND • AMOUNT_USED • FIXED_CHARGE • NONE (for scaled charges) • ANY The default is **ANY**, which lets you add the price tag to multiple types of resource unit.
<PERMITTED>	The services that you can use the price tag with. You can list multiple services, separated by a semicolon. You can use * to match multiple services, for example, **/service/telco/*** to use with services under telco, or ***** to use it with any service.

The following sample creates several new price tags:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ObjectList
  xmlns="http://www.oracle.com/schemas/BusinessConfig"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.oracle.com/schemas/
BusinessConfig ../../../xsd/config_object.xsd">

  <ConfigObject configName="/config/price_tags" configMode="replaceAll">
    <DESCR>Price tags configuration</DESCR>
    <NAME>price_tags</NAME>
    <PRICE_TAGS elem="0">
      <NAME>PRICE1</NAME>
      <DESCR>Dollar Override</DESCR>
      <RULE_TYPE>ANY</RULE_TYPE>
      <RESOURCE_ID>840</RESOURCE_ID>
      <RESOURCE_UNIT>NONE</RESOURCE_UNIT>
      <PERMITTED>/service/email; /service/ip</PERMITTED>
    </PRICE_TAGS>
    <PRICE_TAGS elem="1">
      <NAME>PRICE2</NAME>
      <DESCR>Override seconds charges</DESCR>
      <RULE_TYPE>LIST</RULE_TYPE>
      <CONSTRAINTS>10;20;30;100</CONSTRAINTS>
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
      <RESOURCE_ID>1000095</RESOURCE_ID>
      <RESOURCE_UNIT>MONTH</RESOURCE_UNIT>
       <PERMITTED>/service/*</PERMITTED>
    </PRICE_TAGS>
    <PRICE_TAGS elem="2">
      <NAME>PRICE3</NAME>
      <DESCR>Dollar Override</DESCR>
      <RULE_TYPE>RANGE</RULE_TYPE>
      <CONSTRAINTS>10;200</CONSTRAINTS>
      <RESOURCE_ID>840</RESOURCE_ID>
      <RESOURCE_UNIT>NONE</RESOURCE_UNIT>
      <PERMITTED>/service/telco/*</PERMITTED>
    </PRICE_TAGS>
  </ConfigObject>
</ObjectList>
```

3. Save and close the file.
4. Edit the **load_config** utility's configuration file (*BRM_home***/apps/load_config/pin.conf**) to specify how to connect to the BRM database.
5. Load the **config_price_tags.xml** file into the BRM database:
   a. Switch to the *BRM_home***/apps/load_config** directory.
   b. Run the following command:

      ```
      load_config -vM BRM_home/sys/data/config/config_price_tags.xml
      ```

   See "load_config" in *BRM Developer's Guide* for more information.
6. Stop and restart the Connection Manager (CM).
7. Verify that the price tags were loaded properly by running the following command:

   ```
   load_config -w "price_tags" config_price_tags.xml
   ```

Modifying Price Tag Configuration Objects

You can modify price tag configuration objects in a **config_price_tags.xml** files and load them using the **load_config** utility under certain conditions.

You modify price tag configuration objects the same way you create them. See "Creating Price Tag Configuration Objects".

> ⚠ **Caution**
>
> When you load configuration objects, the **load_config** utility overwrites existing instances of the **/config/price_tags** object. If you are updating price tags or adding new ones, you cannot load new or changed price tags only. You must load the complete set of price tags each time you run the **load_config** utility.

You can modify any fields of price tag configuration objects that have not been added to balance impacts of products or discounts.

Copyright © 2025, 2026, Oracle and/or its affiliates.

For price tags that have been added to products or discounts, you can modify the following fields:

- <DESCR>
- <RULE_TYPE>
- <CONSTRAINTS>

You are responsible for understanding the potential impact that modifying these fields might have on the products and discounts that the price tags have been associated with.

Using Price Tags With Sequential Discounts

BRM supports price tags with sequential discounts, including overriding price tags based on their date. For any date range covered by a price tag, the tag's terms override the discount's default configuration. When a discount includes a price tag that is valid for only part of the billing cycle, BRM splits the billing cycle into segments according to the price tag's start and end times and applies sequential discounts per segment. BRM calculates rating and discounting for segments where the price tag is active, while the rest of the discount is processed without the price tag. This is also true if discounts overlap. BRM creates sub-periods and applies the price tags in each sub-period as the discounts are applied sequentially.

For example, consider a bundle with a monthly charge of $100 for the period January 1 through 31 and these two discounts:

- D1: 10% discount with a price tag valid from January 10 through 19.
- D2: 20% discount with a price tag valid from January 15 through 24.

BRM applies sequential discounts as follows:

- January 1 through 9: D1 and D2 are applied with their default values and no active price tags.
- January 10 through 14: D1 applies with its configured price tag for five days, resulting in a discount of $1.61. That is, [(5 days/31 days x $100) x .10], and D2 applies with its default value and no active price tag.
- January 15 through 19: Both D1 and D2 apply with their configured price tags for five days, resulting in a total discount of $4.51. That is, D1 applies $1.61 first and then D2 applies $2.90. That is, [(5 days/31 days) x ($100 x 0.9) x .20]. Here, 0.9 is the remaining balance after D1.
- January 20 through 24: D1 applies with its default value and no active price tag, and D2 applies with its configured price tag for five days, resulting in a total discount of $3.23. That is, [(5 days/31 days x $100) x .20].
- January 25 through 31: D1 and D2 are applied with their default values and no active price tags.

Copyright © 2025, 2026, Oracle and/or its affiliates.

13

Rating Implementation and Customization

This chapter provides information about implementing and extending the default Oracle Communications Billing and Revenue Management (BRM) rating functionality.

Topics in this document:

- How Rating Works
- Managing Sessions
- Managing Price Lists
- Managing Individual Pricing Objects
- Managing Filter Sets

For general information about rating, see "About Creating a Price List".

How Rating Works

The PCM_OP_ACT_USAGE opcode is the main BRM rating opcode. In addition, subscription opcodes, such as PCM_OP_SUBSCRIPTION_CYCLE_FORWARD, are used for applying fees.

How BRM Rates and Records Usage Events

BRM uses PCM_OP_ACT_USAGE to rate usage events and record them in the BRM database. For example, this opcode is called when an event is generated for a user login session.

> **Note**
>
> Do not call PCM_OP_ACT_USAGE directly.

PCM_OP_ACT_USAGE takes as input an **/account** object POID (Portal Object ID) and a type-only POID that specifies the type of event being recorded.

The PIN_FLD_PRODUCTS array may be included in the input flist to supply or override the list of products that the account owns.

PCM_OP_ACT_USAGE performs the following operations:

1. **Determines whether the event is pre-rated**. If the event is pre-rated, the opcode does not rate the event. However, to support taxation for the following adjustments, disputes, and settlements, this opcode performs additional filtering for pre-rated events:
   - Adjustments at the account level, subscription service level, member service level, and item level
   - Item-level disputes
   - Item-level settlements

Copyright © 2025, 2026, Oracle and/or its affiliates.

This additional filtering consists of checking the event type and performing one of the following activities, as applicable:

- **Adjustments at the account level, subscription service level, member service level:** Gets the adjustment amount and then requests deferred or immediate tax calculation for this amount.
- **Item-level adjustments, disputes, or settlements:** Evaluates the adjustment, dispute, or settlement amount for each event in the bill item and determines the proportionate amount for taxation. It then requests deferred or immediate tax calculation for this amount.

BRM chooses whether to use deferred or immediate taxation for adjustments, disputes, and settlements based on the **tax_now** entry in the Connection Manager (CM) configuration file (**pin.conf**).

For event-level disputes and settlements, PCM_OP_ACT_USAGE also processes notification events that include all the account and balance impact information required to reserve disputed resources and to release them upon settlement.

2. **Adjusts available balances to include reservations**. When PCM_OP_ACT_USAGE is called to *reauthorize* an ongoing prepaid session, an array of the resources currently reserved for the session (the PIN_FLD_RESERVATION_LIST array) is included in the input flist. PCM_OP_ACT_USAGE gets the account balances from which these reservations were made. It then temporarily adds the amount currently reserved for the session to the remaining *unreserved* amount in each retrieved balance. The sum of these two amounts represents the total amount of the resource available to the ongoing session. This amount is used to rate the reauthorization request.

   For example, customer X has a resource balance of 70 free minutes. Currently, 30 of the minutes are reserved for session A, 30 are reserved for session B, and 10 are unreserved. BRM receives a request to extend session A for 30 minutes. Before the request is rated to determine whether customer X's account has sufficient resources for it, PCM_OP_ACT_USAGE adds the free minutes currently reserved for session A (30) to the account's unreserved free minutes (10) to get the total free minutes available to session A (30 + 10 = 40). In this case, the total free minutes available to session A (40) does not cover the reauthorization request, which is for 60 minutes (30 for the initial authorization + 30 for the extension).

3. **Determines whether the event is discountable**. PCM_OP_ACT_USAGE determines whether the event is discountable.

   > **Note**
   >
   > Discountable events include events to which you can apply product discounts, charge sharing, discount sharing, loyalty points, free minutes, and so forth.

   By default, the rating opcodes perform a credit limit check when rating an event. In CALC_ONLY mode, the credit limit check includes the effects of discounts. Because discounts can reduce the balance impact of an event that might otherwise exceed an account's credit limit, the credit limit check for discountable events is deferred to a real-time discounting pipeline.

   To determine whether an event is discountable, PCM_OP_ACT_USAGE checks the usage maps of all the discounts in your system. If at least one map contains the event type of the event being rated, the event is considered discountable.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> PCM_OP_ACT_USAGE evaluates discount usage maps using the data that exists as of the most recent restart of the CM. Instances of event types that are newly added to discount usage maps after the most recent CM restart are therefore not found to be discountable. For example, if you add a new real-time usage event type to an existing discount that does not already include a real-time usage event, events of that type are not found to be discountable. You must restart the CM to make these newly added event types discountable.

If the event is discountable, PCM_OP_ACT_USAGE retrieves all candidate discounts associated with the customer's account. If one or more candidate discounts are returned, the PCM_OP_RATE_EVENT opcode is called with the PCM_OPFLG_RATE_WITH_DISCOUNTING flag set to **True**. This flag turns off the credit limit check performed by the real-time rating opcodes and defers it to the real-time discounting pipeline.

See "Rating Events" for more information about PCM_OP_RATE_EVENT.

4. **Determines how much to charge for the event**. If the calling opcode does not specify a rate, PCM_OP_ACT_USAGE does the following:
   - Reads the **/config/rum** object to generate a ratable usage metric (RUM) candidate for rating the event.
   - Calls PCM_OP_RATE_EVENT to rate the event. See "FM Rate Opcodes Called by PCM_OP_ACT_USAGE".
   - Calls the PCM_OP_RATE_POL_POST_RATING policy opcode to apply any modifications to the **/event/activity** object after rating the event. See "Modifying Rated Events".
5. **Applies the event's balance impact**. PCM_OP_ACT_USAGE applies the balance impact of the event to the account balances as follows:
   - If the account has resources set aside in reservations, the resources are consumed from the reservations.
   - If the account does not have reserved resources, the event balance impact is applied to a balance group in the following order based on what is provided in the input flist:
     - If a bill unit is specified in the input flist, the event balance impact is applied to the default balance group of the bill unit.
     - If a bill unit is not specified, the event balance impact is applied to the specified balance group.
     - If a balance group is not specified, the event balance impact is applied to the balance group of the specified service.
   - Otherwise, the event balance impact is applied to the default account-level balance group.

> **Note**
>
> If the PCM_OPFLG_CALC_ONLY or PIN_ACT_CALC_MAX flag is used, the account balance is not updated.

Copyright © 2025, 2026, Oracle and/or its affiliates.

6. **Records the event**. If no balance impact is associated with the event, PCM_OP_ACT_USAGE checks the Event Record Map configuration object (**/config/event_record_map** cached at CM startup time) against the input event type. If the input event type is configured not to be recorded in BRM, the event is not recorded. Otherwise, the event is recorded.

   PCM_OP_ACT_USAGE performs these tasks:

   - Updates the event fields using information from the specified **/account** object. PCM_OP_ACT_USAGE associates the **/item** object POID with the event and sets the PIN_FLD_CURRENCY, **/payinfo**, **/bill** object POID, and general ledger (G/L) segment ID.

     > **Note**
     >
     > PCM_OP_ACT_USAGE does not record the PIN_FLD_EXTENDED_INFO substruct field in the event object. To record the PIN_FLD_EXTENDED_INFO substruct in the event object for customizations, modify the PCM_OP_RATE_POL_POST_RATING policy opcode.

   - Obtains information about event fields to be cached for event searches and invoicing.
   - Retrieves a list of remittance accounts that must be remitted for the event.

     > **Note**
     >
     > BRM caches remittance specifications. The PCM_OP_REMIT_GET_PROVIDER opcode is called only when the remittance specification cache is non-null.

   - Checks whether the event is included in your system's event notification list. If it is, calls the opcodes associated with the event in the list.

Specifying the Rating Mode

You control how PCM_OP_ACT_USAGE rates and records usage events by using the following flags:

- **PCM_OPFLG_CALC_ONLY**

  When this flag is passed in the opcode call, PCM_OP_ACT_USAGE calculates the amount to charge, factoring in all applicable discounts, but does not apply the amount to the account balance.

- **PIN_ACT_CALC_MAX**

  When PIN_FLD_FLAGS is set to PIN_ACT_CALC_MAX, PCM_OP_ACT_USAGE calculates the maximum quantity that can be consumed based on the event owner's current account balance or reserved amount but does not apply the amount to the account balance.

- **PIN_RATE_FLG_RATE_ONLY**

  When PIN_FLD_FLAGS in the PIN_FLD_BAL_IMPACTS array is set to PIN_RATE_FLG_RATE_ONLY, PCM_OP_ACT_USAGE uses rated, pre-rated, and tax balance impact types to rate the event.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> When PIN_RATE_FLG_RATE_ONLY is set, PCM_OP_ACT_USAGE returns the event's PIN_FLD_BAL_IMPACT array in the output flist.

- **PIN_RATE_FLG_RERATE**

  When the PIN_FLD_FLAGS field in the PIN_FLD_BAL_IMPACTS array is set to PIN_RATE_FLG_RERATE, PCM_OP_ACT_USAGE uses rerated tax balance impact types to rate the event.

> **Note**
>
> When PIN_RATE_FLG_RERATE is set, PCM_OP_ACT_USAGE returns the event's PIN_FLD_BAL_IMPACT array in the output flist.

- **PIN_RATE_FLG_OVERRIDE_CREDIT_LIMIT**

  When PIN_FLD_FLAGS in the PIN_FLD_BAL_IMPACTS array is set to PIN_RATE_FLG_OVERRIDE_CREDIT_LIMIT, PCM_OP_ACT_USAGE does not check the credit limit.

FM Rate Opcodes Called by PCM_OP_ACT_USAGE

To rate events, PCM_OP_ACT_USAGE calls the following opcodes:

- PCM_OP_RATE_EVENT. See "Rating Events".
- PCM_OP_RATE_GET_PRODLIST. See "Retrieving Product Lists".

Rating Events

To rate an event, PCM_OP_RATE_EVENT does the following:

- When the optional time-stamp field PIN_FLD_WHEN_T is present in the PCM_OP_RATE_EVENT input flist, PCM_OP_RATE_EVENT searches for the price list that is valid at the time specified in the field. It uses the price list to rate the event.
- PCM_OP_RATE_EVENT sets the optional PIN_FLD_VALID_TO field for CALC_MAX and CALC_ONLY operations. This field is used to limit prepaid product authorizations to a specific time period for non-duration RUMs.
- If a product or discount starts on first usage and its validity period has not been set, selects the product or discount as a candidate for rating if the purchase time is equal to or earlier than the event time. The opcode returns the first-usage products that are used to rate the event in the PIN_FLD_FIRST_USAGE_PRODUCTS array in the output flist.
- If rating impacts a balance whose validity period has not yet been set (such as noncurrency balances that start on first usage or relative to when they are granted), calculates the resource balance validity period. If resource validity end time is restricted to the granting product's or discount's end time, this opcode compares the calculated end time with the product's or discount's end time and returns the earlier of the two in the output flist.
- Locates any rollover objects for events and returns details of the rollover object to the calling opcode.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Determines whether it should perform a credit limit check. PCM_OP_RATE_EVENT performs a credit check unless:
  - The event is discountable or involves multiple RUMs. In CALC_ONLY mode, the credit limit check is deferred to the real-time pipeline.
  - The event being rated includes the PIN_RATE_FLG_NO_CREDIT_LIMIT_CHECK flag, the applicable rate is a credit limit override rate, or the event is a refund event. (The PIN_RATE_FLG_NO_CREDIT_LIMIT_CHECK flag is set if the event associated with the plan does not use a credit limit. You specify to not use a credit limit in Pricing Center when you set up a plan.)

  PCM_OP_RATE_EVENT sets a value in the PIN_FLD_CHECK_CREDIT_LIMIT field in its output flist to indicate whether a credit limit check should occur, where it should occur, and how the real-time discounting pipeline should handle the event if the credit limit occurs there. Flag values can be summed.
  - PIN_RATE_CREDIT_LIMIT_NEEDED (**0x1000**) indicates that a credit limit check should be performed.
  - PIN_RATE_CREDIT_LIMIT_LOCATION_CRE (**0x100**) indicates that the credit limit check is handled by PCM_OP_RATE_EVENT.
  - PIN_RATE_CREDIT_LIMIT_LOCATION_RTP (**0x200**) indicates that the credit limit check is deferred to the real-time discounting pipeline.
  - PIN_RATE_NO_CREDIT_LIMIT_DISCOUNT_ONLY (**0x0**): Indicates, for events that are processed by the real-time discounting pipeline, that no credit limit check should occur. This flag is used for events sent to the pipeline for discounting only, typically when rating in normal, as opposed to CALC_ONLY, mode.
  - PIN_RATE_CREDIT_LIMIT_CHECK (**0x1**): Indicates, for events that are processed by the real-time discounting pipeline, that a credit limit check should be performed by the FCT_CreditLimitCheck module. This flag is typically used for prepaid authorization. If the full amount cannot be authorized, FCT_CreditLimitCheck determines the maximum amount that can be authorized.
- If the event qualifies for a discount, sends the event to a real-time discounting pipeline.
- Calls the PCM_OP_RATE_TAX_CALC opcode to calculate taxes.
- When rating cycle forward events, calculates fees and refunds based on scale values, proration settings, and rates. The opcode uses the values of PIN_FLD_SCALE and PIN_FLD_ORIGINAL_SCALE passed to it by Subscription Manager to determine the scale to apply. These values are set differently depending on whether they are used for applying cycle fees or refunding them.

  The opcode calculates the cycle fee or refund based on the following formula:

  PIN_FLD_SCALE * PIN_FLD_ORIGINAL_SCALE * Rate

  The opcode retains the value of PIN_FLD_ORIGINAL_SCALE that is passed to it, but it sometimes changes the value of PIN_FLD_SCALE depending on the proration setting.

  To calculate cycle fees, the opcode sets PIN_FLD_SCALE to the following values:
  - When purchase proration is set to **Do not charge for this cycle**, PIN_FLD_SCALE is set to **0**.
  - When purchase proration is set to **Charge for the entire cycle**, PIN_FLD_SCALE is set to **1**.
  - When purchase proration is set to **Calculate the charge based on the amount used**, PIN_FLD_SCALE retains the value passed to the opcode by Subscription Manager.

  To calculate cycle fee refunds, the opcode sets PIN_FLD_SCALE to the following values:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- When cancel proration is set to **Do not charge for this cycle**, PIN_FLD_SCALE is set to **1**.
- When cancel proration is set to **Charge for the entire cycle**, PIN_FLD_SCALE is set to **0**.
- When cancel proration is set to **Calculate the charge based on the amount used**, PIN_FLD_SCALE retains the value passed to the opcode by Subscription Manager.

The opcode includes PIN_FLD_SCALE and PIN_FLD_ORIGINAL_SCALE in its output flist. These fields are stored in the **/event/billing/product/fee/cycle** object.

Retrieving Product Lists

To get the list of products for an event, PCM_OP_ACT_USAGE calls the PCM_OP_RATE_GET_PRODLIST opcode. This opcode gets a list of purchased products (**/purchased_product** objects) for an account based on the combination of service and event type in its input flist. It returns a list of base products and valid customized products.

In its initial search for products, the opcode uses the event object and optional service object in the flist to determine which products it retrieves:

- If the input flist does not include a service object POID, the opcode finds all purchased products for the account.
- If a service is included in the input flist, the opcode selects products that apply to the specified service and the event type in the flist. For example, if the event type is **/event/session/telco/gsm** and the service is **/service/telco/gsm/telephony**, the opcode finds products that apply to GSM usage.

After retrieving the list of products, the opcode checks if any of the products are overridden by a customized product that is currently valid. If a product is overridden by a valid customized product, it is removed from the list and not returned by the opcode. The valid customized product is included in the list instead.

If a product is retrieved and its validity period is not yet initialized, information about the product's purchase, cycle, and usage start and end times is returned in the purchase, cycle, and usage *_START_DETAILS and *_END_DETAILS fields respectively.

Generating Ratable Usage Metrics

Use the PCM_OP_ACT_GET_CANDIDATE_RUMS opcode to generate a ratable usage metric (RUM) that can be used to rate the event. A candidate RUM specifies the name, quantity, and unit of measurement for each resource used in the event. For example, an IP fax event's RUM might include the information in Table 13-1:

Table 13-1 Generating Ratable Usage Metrics

Name	Quantity	Unit
duration	2	minutes
page_count	10	pages
byte_count	9568	bytes

PCM_OP_ACT_GET_CANDIDATE_RUMS performs the following operations:

1. Calls the PCM_OP_ACT_POL_SPEC_CANDIDATE_RUMS policy opcode to calculate RUMs. See "Customizing How to Calculate RUMs".

Copyright © 2025, 2026, Oracle and/or its affiliates.

2. Reads the **/config/rum** object to retrieve the list of valid RUMs for the specified event type.
3. Combines the configuration data with the event data to produce a RUM candidate for rating the event.
4. Returns the following data about each RUM:
   - RUM name
   - Quantity to rate
   - Unit (seconds, minutes, bytes)

About Calculating the Maximum Available Usage

BRM uses the PCM_OP_ACT_CALC_MAX_USAGE opcode to calculate the maximum available usage based on one of the following:

- The account's credit limit and current account balance.
- The amount reserved for the session.

For example, this opcode is called by the PCM_OP_ACT_AUTHORIZE opcode to calculate the maximum possible duration of a call to prevent prepaid customers from exceeding their current account balance.

PCM_OP_ACT_CALC_MAX_USAGE performs the following operations:

1. Specifies a huge quantity for the event RUM.
2. Calls PCM_OP_ACT_USAGE with PIN_FLD_FLAG set to PIN_ACT_CALC_MAX to rate and return the maximum quantity available based on the current balance or the reservation amount. See "How BRM Rates and Records Usage Events".
3. Returns PIN_FLD_QUANTITY set to one of the following:
   - The maximum usage allowed based on the user's available balance and credit limit or reserved amount.
   - **-1** to indicate unlimited usage. This value is returned only when *all* of the account's available resources can be consumed.

> **Note**
>
> By default, for a duration-type RUM, PCM_OP_ACT_CALC_MAX_USAGE limits the maximum duration to 24 hours. For example, if a user's available balance enables the user to make a call longer than 24 hours, this opcode returns **-1**. To configure the duration maximum, set the **fm_act max_qty_for_duration** entry to a value greater than 24 (hours) in the CM configuration file.

This opcode propagates any errors returned from PCM_OP_ACT_USAGE.

About Currency Conversion

During rating, currency conversion is handled by the following opcodes:

- The PCM_OP_BILL_CURRENCY_CONVERT_AMOUNTS opcode converts the currency according to the conversion rate defined in the **/config/currency/conversionrates** object. You can set multiple time periods for conversion rates.

Copyright © 2025, 2026, Oracle and/or its affiliates.

This opcode fails if the specified time is not within the time range or if the source or destination currency is invalid. The ERROR_NOT_FOUND error code is returned.

- The PCM_OP_BILL_CURRENCY_QUERY_CONVERSION_RATES opcode retrieves the conversion rates from the **/config/currency/conversionrates** object. This opcode is called by the PCM_OP_BILL_CONVERT_AMOUNTS opcode.

  This opcode fails if no conversion rate is specified between the source and destination currency types. The ERROR_NOT_FOUND error code is returned.

About Applying Cycle Forward Fees

To apply a cycle forward fee to a balance group, use the PCM_OP_SUBSCRIPTION_CYCLE_FORWARD opcode.

This opcode is called to charge or refund cycle forward fees (for example, when a product or discount is purchased, canceled, activated, or inactivated).

PCM_OP_SUBSCRIPTION_CYCLE_FORWARD performs the following operations:

1. When a product or discount is *purchased* or *activated* during the cycle, determines scale values that are used by real-time rating to prorate the cycle forward fee amount to be charged.
2. When a product or discount is *canceled* or *inactivated* during the cycle, determines scale values that are used by real-time rating to prorate the cycle forward fee amount to be refunded.
3. Uses customized products when valid to calculate fees or refunds. If a customized product is valid for only part of a cycle, it contributes toward the total charge or refund based on the length of its validity.
4. When a rate change is scheduled for the *next* or *current* cycle, PCM_OP_SUBSCRIPTION_CYCLE_FORWARD gets a list of rates and the period in which they are applicable and calculates the correct charges.
5. Calculates and sets CYCLE_FEE_START_T and CYCLE_FEE_END_T to the next period, if applicable.
6. After all the products and discounts and their cycle forward rates are determined, it sends the information to PCM_OP_ACT_USAGE to rate and apply the charges.

> **Note**
>
> If the product or discount starts on first usage (when the customer first uses the product or discount) and its validity period has not yet been set, PCM_OP_SUBSCRIPTION_CYCLE_FORWARD does not call PCM_OP_ACT_USAGE, and cycle fees are not applied.

7. For auditing purposes, it creates **/event/billing/product/fee/cycle/cycle_forward**_*type* objects, where *type* is the frequency of the cycle forward charge (for example, daily, weekly, monthly, bimonthly, quarterly, semiannual, or annual).
8. If successful, it returns the POIDs of the **/account** object and the **/event/billing/product/fee/cycle/cycle_forward**_*type* event.

About Applying Cycle Arrears Fees

To apply a cycle arrears fee, use the PCM_OP_SUBSCRIPTION_CYCLE_ARREARS opcode.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> Cycle arrears fees are applied only for a single month.

PCM_OP_SUBSCRIPTION_CYCLE_ARREARS performs the following operations:

1. When products or discounts with cycle fees are canceled or inactivated during a cycle, calculates the scale to prorate the cycle arrears fee amount to be charged.
2. When a rate change occurred in the *previous* or *next* cycle, gets a list of rates and the period in which they are applicable and calculates the correct charges.
3. Uses customized products when valid to calculate fees or refunds. If a customized product is valid for only part of a cycle, it contributes a portion of the total charge or refund based on the length of its validity.
4. After all the product and discount cycle arrears rates are determined, sends the information to PCM_OP_ACT_USAGE to rate and apply the charges.

> **Note**
>
> If the product or discount starts on first usage (when the customer first uses the product or discount) and its validity period has not yet been set, PCM_OP_SUBSCRIPTION_CYCLE_ARREARS does not call PCM_OP_ACT_USAGE, and cycle fees are not applied.

5. Creates the **/event/billing/product/fee/cycle/cycle_arrears** event for auditing purposes.
6. If successful, returns the POIDs of the **/account** object and the **/event/billing/product/fee/cycle/cycle_arrears** event.

Customizing the Cycle Interval for Products

To customize the time interval for applying cycle forward and cycle arrears fees for a specified product, use the PCM_OP_SUBSCRIPTION_POL_SPEC_CYCLE_FEE_INTERVAL policy opcode.

This policy opcode is called by PCM_OP_SUBSCRIPTION_CYCLE_FORWARD and PCM_OP_SUBSCRIPTION_CYCLE_ARREARS. The type of cycle event is passed in the PIN_FLD_SCOPE field in the input flist.

By default, this policy opcode is an empty hook to facilitate customization of the cycle forward and cycle arrears start (CHARGED_FROM_T) and end dates (CHARGED_TO_T) for a specific product. The start and end dates provided are used by rating opcodes to calculate the scale to determine the cycle fee amount to charge or refund.

The PIN_FLD_SCALE value in the input and output flists is the original charge scale and is used only to calculate the refund scale.

For example, if a product is purchased on April 1, 2009, with a monthly cycle forward fee of $30 and the purchase, usage, and cycle end dates are set to April 20, 2009, the cycle fee amount is based on the scale for the period April 1, 2009, to April 20, 2009 (20 days) divided by the unit interval from April 1, 2009, to April 30, 2009 (30 days). The cycle fee charged is **20/30 * $30**, or $20.

Copyright © 2025, 2026, Oracle and/or its affiliates.

If the product is canceled on April 15, 2009, the refund amount is based on the scale for the period April 15, 2009, to April 20, 2009 (5 days) divided by the unit interval from April 1, 2009, to April 20, 2009 (20 days). Because the refund amount is refunded from the charged amount ($20), the refund is **5/20 * (20/30 * $30)**, or $5. Here, the scale value 20/30 is the original charge scale or the period the product was valid during the cycle.

To change the scale (for example if you do not want to refund the full amount), change the start and end dates. The refund scale is calculated based on the dates that you provide.

About Restricting the End Time of Granted Resources That Start on First Usage

> **Note**
>
> This is an optional task that you perform only when you configure the validity periods of granted resources to start on first usage. For more information, see "About Balance Impacts That Become Valid on First Usage".

You can configure BRM to automatically restrict the validity period of granted resources to end no later than the end time of the product or discount that grants the resource.

When you configure resource validity to start on first usage and end on a date relative to the start date (when first usage occurs), you cannot know the actual end date when setting up the rate plan. Restricting the resource end time to the product or discount end time ensures that the resource cannot continue to be consumed after the product or discount expires.

> **Note**
>
> When a product or discount is *canceled*, the validity period end time of resources granted by that product or discount is set to the time of the cancellation.

When you restrict resource validity end time, BRM sets the end time to the product or discount end time at the time of the grant. When the customer consumes the granted resource for the first time, the relative end time is calculated:

- If the calculated end time is *later* than the granting product or discount end time, the resource validity period uses the product or discount end time.
- If the calculated end time is *earlier* than the granting product or discount end time, the resource validity period uses the calculated end time.

> **Note**
>
> If you restrict resource validity end time, you must do so for both real-time rating and pipeline rating.

To restrict resource validity end time, see "Configuring Real-Time Rating to Restrict Resource Validity End Time".

Copyright © 2025, 2026, Oracle and/or its affiliates.

Configuring Real-Time Rating to Restrict Resource Validity End Time

In real-time rating, you restrict resource validity end time to the end time of the product or discount that grants the resource by modifying a field in the **multi_bal** instance of the **/config/business_params** object.

You modify the **/config/business_params** object by using the **pin_bus_params** utility.

To restrict resource validity end times to product or discount end times:

1. Run the following command, which creates an editable XML file from the **multi_bal** instance of the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsMultiBal bus_params_multi_bal.xml
   ```

   This command creates the XML file named **bus_params_multi_bal.xml.out** in your working directory. If you do not want this file in your working directory, specify the path as part of the file name.

2. Search the XML file for following line:

   ```
   <RestrictResourceValidityToOffer>FALSE</RestrictResourceValidityToOffer>
   ```

3. Change **FALSE** to **TRUE**.

   > **Note**
   >
   > BRM uses the XML in this file to overwrite the existing **multi_bal** instance of the **/config/business_params** object. If you delete or modify any other parameters in the file, these changes affect the associated aspects of the BRM multi-balance configuration.

4. Save the file and rename it from **bus_params_multi_bal.xml.out** to **bus_params_multi_bal.xml**.

5. Run the following command, which loads the change into the **/config/business_params** object:

   ```
   pin_bus_params bus_params_multi_bal.xml
   ```

   Run this command from the *BRM_home*/**sys/data/config** directory (where *BRM_home* is the directory in which BRM is installed), which includes support files used by the utility.

6. Read the object with the **testnap** utility or Object Browser to verify that all fields are correct.

7. Stop and restart the CM.

About Applying Folds

For background information about folds, see "About Fold Events".

Use the PCM_OP_SUBSCRIPTION_CYCLE_FOLD opcode to apply balance impacts for fold events.

PCM_OP_SUBSCRIPTION_CYCLE_FOLD performs the following operations:

Copyright © 2025, 2026, Oracle and/or its affiliates.

1. Calls the PCM_OP_SUBSCRIPTION_POL_SPEC_FOLD policy opcode to get the order in which to fold the account's resources. By default, folds are applied in ascending order of the resource IDs.
2. Checks the value of the resource's PIN_FLD_APPLY_MODE field in the **/config/beid** object to see whether the resource is specified to be rolled over. For more information, see "Specifying Which Resources to Fold".
3. Applies folds for all of the account's balance groups and also for each resource in the balance group that is specified to be rolled over. However, if a resource ID is specified in the input flist, only that resource is folded.
4. After applying the balance impacts, creates **/event/billing/cycle/fold** event for auditing purposes.
5. If successful, returns the POIDs of the **/account** object and the **/event/billing/product/fee/cycle/fold** event.

About Applying Folds Only For Account-Level Products

If a fold is configured at account level, the fold is applicable to all the services in the account. The fold is then applied to the first service that is retrieved. If the resource balance is non-zero after applying the fold to the first service, the fold is applied again to the next service until the resource balance is zero.

To apply the fold only to account-level products, you can configure an account-level balance group to track the resource that you want to fold. The fold is then applied to only the products associated with the account-level balance group and not to all the services in the account.

Customizing How Folds Are Applied

To customize how folds are applied, use the PCM_OP_SUBSCRIPTION_POL_PRE_FOLD policy opcode.

By default, this policy opcode is an empty hook provided to facilitate any customization before folding the currency or noncurrency resources. For example, when billing is run, this policy opcode is called to verify that the **pin_cycle_fees** utility has applied cycle fees to an account.

Customizing the Order to Apply Folds

To customize the order in which folds are applied, use the PCM_OP_SUBSCRIPTION_POL_SPEC_FOLD policy opcode.

By default, resources are folded in ascending order based on the resource ID. This policy opcode enables you to change the order in which resources are folded.

For example, you can fold resources in descending order of the resource IDs. To do this, sort the PIN_FLD_BALANCES array and return the sorted array.

The following example shows the PIN_FLIST_SORT statement for sorting the balances array in descending order:

```
*out_flistp = PIN_FLIST_COPY(i_flistp, ebufp);
s_flistp = PIN_FLIST_CREATE(ebufp);
PIN_FLIST_ELEM_SET(s_flistp, (void *)NULL, PIN_FLD_BALANCES, PIN_ELEMID_ANY, ebufp);

PIN_FLIST_SORT(*out_flistp, s_flistp, 1, ebufp);
```

This policy opcode returns the contents of the input flist including the POID of the **/balance_group** object and the balance array sorted in ascending order.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Specifying Which Resources to Fold

You can specify which resources are impacted by fold events. This enables you to exclude folding resources that do not need to be folded.

To specify the resources to fold, you set the PIN_FLD_APPLY_MODE parameter for the resources in the resource configuration file (*BRM_home*/**sys/data/pricing/example/pin_beid**).

The PIN_FLD_APPLY_MODE parameter can take the following values:

- **1** indicates the resource will be folded, if appropriate.
- **0** indicates the resource will not be folded.

By default, folds are enabled for all resources in the **pin_beid** file.

After modifying the **pin_beid** file, load the contents of the file into the **/config/beid** object in the BRM database by running the **load_pin_beid** utility.

For more information, see the **pin_beid** file and "load_pin_beid".

Customizing Which Resources to Fold When Products Are Canceled

The PCM_OP_SUBSCRIPTION_POL_PREP_FOLD policy opcode prepares the list of resources that must be folded when a product is canceled. This opcode is called when a product is canceled, and it performs the following functions:

- Checks the canceled product.
- Creates a list of resources affected by the cancellation that must be folded.
- Calls PCM_OP_SUBSCRIPTION_CYCLE_FOLD and passes the list of resources as the input.

If you do not want to fold resources after a product is canceled, customize this policy opcode by removing or commenting out the code in the **fm_subscription_pol_prep_fold.c** file.

Assigning Rate Plans and Impact Categories to Events

Use the PCM_OP_ACT_POL_SPEC_RATES policy opcode to map an event to a rate plan and impact category.

> **Note**
>
> Ensure that your rate plan structure uses custom event analysis for the event.

The opcode that calls the PCM_OP_ACT_POL_SPEC_RATES policy opcode passes in the inherited information for an event. For example, to rate a password, the PCM_OP_CUST_SET_PASSWD opcode specifies the inherited password information. The PCM_OP_ACT_POL_SPEC_RATES policy opcode specifies that the password is to be rated.

Though most custom rating functionality is now in the rate plan selector, some special cases require you to use the PCM_OP_ACT_POL_SPEC_RATES policy opcode. In such a case, you must:

Copyright © 2025, 2026, Oracle and/or its affiliates.

1. Map your custom events to BRM opcodes in the *BRM_home*/**sys/data/config/pin_spec_rates** file, and then load the file into the BRM database by using the **load_pin_spec_rates** utility.
2. Define your custom impact categories in the *BRM_home*/**sys/data/config/pin_impact_category** file, and then load the file into the BRM database by using the **load_pin_impact_category** utility.

This configures the PCM_OP_ACT_POL_SPEC_RATES policy opcode and the rate plan selector to return an event's rate plan and impact category to real-time rating.

Rating an Event Based on Extended Rating Attributes

An event can be rated based on extended rating attributes (ERAs) by using either the usage type or the ERA label, as follows:

- **Usage type**: You can select a specific rate plan and impact category based on the usage type of an event, such as ERAs. For example, if the usage type for an event is Friends and Family, you can rate that event with a special rate of $1.00 a minute.
- **ERA label**: You can select a specific rate plan based on the ERA label. A label is a separate list of ERA values. You can have multiple labels for a single ERA and use a rate plan selector to choose a different rate based on the label.

  For example, you can have a friends and family ERA with two labels, MYFRIENDS and MYFAMILY, and use the rate plan selector to use a different rate for each list.

For more information on ERAs, see "About Extended Rating Attributes".

To rate an event by using special rates based on an ERA, PCM_OP_ACT_USAGE calls the PCM_OP_RATE_POL_PRE_RATING policy opcode before calling PCM_OP_RATE_EVENT.

The PCM_OP_RATE_POL_PRE_RATING policy opcode calls the PCM_OP_RATE_POL_PROCESS_ERAS policy opcode, which in turn calls the PCM_OP_RATE_GET_ERAS opcode.

PCM_OP_RATE_GET_ERAS retrieves the valid ERAs for an event by performing the following operations:

1. Gets the service and account data for the event from the input flist.
2. Finds the ERA information in the associated **/profile/serv_extrating** or **/profile/acct_extrating** object. The ERA information includes the ERA name, label names, label values, and validity dates.

   > **Note**
   >
   > By default, BRM includes all ERAs in this process, but you can improve real-time rating performance by configuring BRM to omit checking for account-level ERAs, service-level ERAs, or both during rating.

3. Identifies profile sharing groups that the service or subscription service for the event belongs to. If membership is found and the group is active, the opcode retrieves the ERA information.
4. For shared profile groups, finds ERA information in the associated **/group/sharing/profiles** object. The ERA information includes the ERA name, label names, label values, and validity dates.

Copyright © 2025, 2026, Oracle and/or its affiliates.

5. Checks whether the ERA is valid for the event start or end time, based on the configuration.

   > **Note**
   >
   > The configuration for using event start time for the validity check is determined by the **use_prods_based_on_start_t** entry in the CM configuration file. See "Using Event Start Time to Determine a Product's Validity".

6. If the event time is less than the effective time of any of the profiles, searches for that profile in the audit tables to get the correct data matching the event time.

   If the ERA is found to be not valid, the opcode skips the remaining steps.

7. If the service has a subscription object, reads the service profiles of that subscription object. If the same ERA is found in the service and subscription service, the service ERA takes precedence.

8. Returns the names of valid ERAs to the PCM_OP_RATE_POL_PROCESS_ERAS policy opcode.

   The PCM_OP_RATE_POL_PROCESS_ERAS policy opcode uses the data retrieved by PCM_OP_RATE_GET_ERAS to do the following:

   - Compare certain predefined event fields to the values for the ERAs found in the service or account.

     The specific fields compared depend on the service. By default, the policy opcode is configured to work with a GSM telco service and to compare values in the PIN_FLD_CALLING_FROM and PIN_FLD_CALLED_TO fields.

     The policy opcode can be easily configured to work with a GPRS service. In that case, it compares values in the PIN_FLD_ANI and PIN_FLD_DN fields.

     > **Note**
     >
     > Using the PCM_OP_RATE_POL_PROCESS_ERAS policy opcode with services other than GSM and GPRS requires customization. See "Customizing Real-Time Rating for Friends and Family ERAs".

   - Populate the PIN_FLD_USAGE_TYPE field with the names of the valid ERAs.

     If multiple ERAs are found for service and account profiles, the ERAs can be combined in the PIN_FLD_USAGE_TYPE field. For example, if Closed User Group (CUG) and Friends & Family (FF) are the qualified ERAs for the event, PIN_FLD_USAGE_TYPE can be set to FF_CUG. By default, PIN_FLD_USAGE_TYPE is populated based on the following qualified ERAs:

     – Friends and Family (FF)
     – Closed User Group (CUG)
     – FF + CUG (FF_CUG)

     You can customize the policy opcode to add other ERAs.

   - Populate the PIN_FLD_PROFILE_LABEL_LIST field with ERA label names. Multiple names are separated by a comma, unless you specified a different separator in the opcode and the rate plan selector.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Return the output to the PCM_OP_RATE_POL_PRE_RATING policy opcode.

9. The PCM_OP_RATE_POL_PRE_RATING policy opcode returns the output flist with the PIN_FLD_USAGE_TYPE and PIN_FLD_PROFILE_LABEL_LIST fields to the calling opcode.
10. The event is then rated or discounted based on the ERA value in either PIN_FLD_USAGE_TYPE or PIN_FLD_PROFILE_LABEL_LIST.

    If there are multiple valid ERAs and a rate plan selector is being used for the event, the rating or discounting opcode uses the rate plan selector to determine which ERA should be used for rating or discounting. The rate plan selector would use either EVENT.PIN_FLD_USAGE_TYPE or EVENT.PIN_FLD_PROFILE_LABEL_LIST to determine a rate based on the ERA.

Modifying Rated Events

Use the PCM_OP_RATE_POL_POST_RATING policy opcode to modify rated **/event** objects (for example, change the G/L ID of an event). This policy opcode is called by PCM_OP_ACT_USAGE.

The input flist matches the **/event** object that you are modifying. The output flist contains the event field to be changed in the rated object.

The following example shows how to change the G/L ID of an event:

```
STORABLE CLASS /event {
DESCR = "Objects of the event class are created to record the various "
"system-initated and user-initiated events. The events are "
"either related to a specific service or to an account. The "
"event includes generic information such as start and end "
"times as well the various charges that are incurred by the "
"account due to this event.";
SEQ_START = "1";
INT32 PIN_FLD_GL_ID {
DESCR = "GLID associated with this balance impact. "
"Don't care if 0.";
ORDER = 0;
CREATE = Optional;
MODIFY = Writable;
}
}                          CREATE = Optional;
MODIFY = Writable;
```

> **⚠ Caution**
>
> Before calling the PCM_OP_RATE_POL_POST_RATING opcode, the PCM_OP_ACT_USAGE opcode stores the balance of a rated event in a balance group cache rather than in the BRM database. If you customize the PCM_OP_RATE_POL_POST_RATING opcode to call a standard opcode to change the balance of the rated event directly in the database or in a different cache, the balance group cache from which the PCM_OP_ACT_USAGE opcode fetches the final balance impact of the event may not reflect the change made by the PCM_OP_RATE_POL_POST_RATING opcode.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Customizing How to Calculate RUMs

Use the PCM_OP_ACT_POL_SPEC_CANDIDATE_RUMS policy opcode to define the customized policy to specify the candidate RUM.

A candidate RUM specifies the name, quantity, and unit of measurement for each resource used in the event. For example, an IP fax event's RUM might have the information in Table 13-2:

Table 13-2 Example

Name	Quantity	Unit
duration	2	minutes
page_count	10	pages
byte_count	9568	bytes

By default, the PCM_OP_ACT_POL_SPEC_CANDIDATE_RUMS policy opcode copies PIN_FLD_POID and the PIN_FLD_CANDIDATE_RUMS array to the output flist.

You can change the rules to calculate data by using **/config/rum**. By default, the PCM_OP_ACT_POL_SPEC_CANDIDATE_RUMS policy opcode supports only simple-arithmetic expressions (addition, subtraction, multiplication, and division), but you can add more complex rules to get the event quantity.

If you add a candidate RUM that has the same name as any RUM in the input flist, drop the one from the input flist to ensure that the RUM names are unique.

Rating and Recording Activity Events

BRM uses the PCM_OP_ACT_ACTIVITY opcode to record and rate activity events. This enables BRM to record any type of activity event and any details specific to the event type. Any type of user action can be recorded as an activity event, but it is especially designed to represent events that occur at a single point in time. All activity events are recorded in the system simultaneously.

> **Note**
>
> For events related to customer service usage over a period, the PCM_OP_ACT_START_SESSION and PCM_OP_ACT_LOGIN opcodes are called. For more information on the PCM_OP_ACT_START_SESSION and PCM_OP_ACT_LOGIN opcodes, see *BRM Developer's Reference*.

PCM_OP_ACT_ACTIVITY records events for either **/account** or **/service** objects. The **/account** POID must be specified in both cases.

- When an **/account** POID is used alone, PCM_OP_ACT_ACTIVITY records the event for an **/account** object.
- When both an **/account** POID and a **/service** POID are specified, PCM_OP_ACT_ACTIVITY records the event for a **/service** object.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- When a **NULL /service** POID is specified, PCM_OP_ACT_ACTIVITY records an **/account** event.

PCM_OP_ACT_ACTIVITY records event details in the following ways:

- Using the generic **/event/activity** object, the PIN_FLD_DESCR field specifies details of the event in ASCII text format.
- If this is insufficient, an inherited object type can be created by the user from the **/event/activity** object and additional fields added to describe a specific type of event in more detail. When an event of that type occurs, the input flist to this operation must use the PIN_FLD_OBJ_TYPE field to specify the type of event object to create and to include the detailed information fields in the PIN_FLD_INHERITED_INFO substructure. This enables any amount of detail to be recorded for any number of event types.

> **Note**
>
> The PIN_FLD_INHERITED_INFO feature requires you to supply a new storable class definition. You determine the fields necessary in the new inherited storable class type and define them by using PIN_MAKE_FLD.

BRM controls what PCM_OP_ACT_ACTIVITY returns by using the following flags:

- **PCM_OPFLG_READ_RESULT**

  When this flag is set, PCM_OP_ACT_ACTIVITY returns all fields in the event object, in addition to the POID.

- **PCM_OPFLG_CALC_ONLY**
  - When this flag is set, PCM_OP_ACT_ACTIVITY only calculates the rated amount and does not record the event in the BRM database. The opcode returns the fields that would have been used to create the event object. No fields in the database are changed, and the event object is not actually created.
  - When this flag is *not* set, PCM_OP_ACT_ACTIVITY creates the **/event/activity** object to record details of the operation.

Managing Sessions

BRM stores information about sessions in **/event/session** objects in the BRM database.

A session event differs from an activity event in that the start and end of the session event are recorded separately. A session is designed to efficiently track a customer connecting to some type of service for a period of time. For events that occur at a single point in time, the PCM_OP_ACT_ACTIVITY opcode is called. For more information on the PCM_OP_ACT_ACTIVITY opcode, see *BRM Developer's Reference*.

You use the following Activity standard opcodes to manage sessions:

- PCM_OP_ACT_LOGIN. See "Starting Sessions".
- PCM_OP_ACT_START_SESSION. See "Recording the Start of a Session".
- PCM_OP_ACT_UPDATE_SESSION. See "Updating a Session Event".
- PCM_OP_ACT_LOGOUT. See "Recording the End of a Session".
- PCM_OP_ACT_END_SESSION. See "Rating and Recording Session Events".
- PCM_OP_ACT_LOAD_SESSION. See "Loading Sessions in Real Time".

Copyright © 2025, 2026, Oracle and/or its affiliates.

Starting Sessions

Use the PCM_OP_ACT_LOGIN opcode to start a customer session.

PCM_OP_ACT_LOGIN performs the following operations:

1. Calls the PCM_OP_ACT_FIND_VERIFY opcode to verify the customer's identity.
2. Calls the PCM_OP_ACT_START_SESSION opcode to start a login session. See "Recording the Start of a Session".
3. Returns an flist with the user's **/account** POID.

This opcode is usually called by PCM_CONNECT, but it is also called directly. If called by PCM_CONNECT, this opcode ignores the database number in the application's **pin.conf** file and searches all schemas for the account.

Recording the Start of a Session

Use the PCM_OP_ACT_START_SESSION opcode to record the start of a session event.

This opcode records any type of session event object including details specific to the event type. This opcode does not perform rating. Fees for the session event are calculated and applied when the session ends.

Sessions can be started for either **/account** or **/service** objects. The **/account** object must be specified for both cases.

- When the **/account** object is used alone, PCM_OP_ACT_START_SESSION starts a session for the **/account** object.
- When both **/account** and **/service** object POIDs are specified, PCM_OP_ACT_START_SESSION starts a session for the **/service** object.
- When a **NULL /service** object is specified, PCM_OP_ACT_START_SESSION starts a session for the **/account** object.

Session details can be recorded in the following ways:

- In the generic **/event/session** object, the PIN_FLD_DESCR field specifies details of the session in a text format.
- If that is insufficient, you can create an inherited object type from the **/event/session** object and add fields to describe a specific type of session in more detail.

  When a session of the specified type occurs, the input flist to PCM_OP_ACT_START_SESSION can specify the type of event object to create with the PIN_OBJ_TYPE field and include the detailed information fields in the PIN_FLD_INHERITED_INFO substructure. This enables any amount of detail to be recorded for any number of session types.

  Within the PIN_FLD_INHERITED_INFO array, the program supplies a new storable class definition. You determine the fields necessary in the new inherited storable class type and define them by using PIN_MAKE_FLD. This procedure is explained in the **pcm.h** file. PCM_OP_ACT_START_SESSION then automatically creates storable class definitions and supplies updated release files containing the new storable class definitions.

BRM controls how PCM_OP_ACT_START_SESSION records the start of a session event by using the following flags:

- **PCM_OPFLG_READ_RESULT**

Copyright © 2025, 2026, Oracle and/or its affiliates.

When this flag is set, PCM_OP_ACT_START_SESSION returns all fields in the event object in addition to the **/account** POID.

- **PCM_OPFLG_CALC_ONLY**
  - When this flag is set, PCM_OP_ACT_START_SESSION returns the fields that would have been used to create the event object. No fields in the database are changed, and the event object is not actually created.
  - When the flag is *not* set, PCM_OP_ACT_START_SESSION creates an **/event/session** object to record the details of the session event.

Updating a Session Event

Use the PCM_OP_ACT_UPDATE_SESSION opcode to update information in a session event.

> **Note**
>
> The session event object must already have been created by a call to PCM_OP_ACT_START_SESSION or by an opcode that calls that opcode.

This opcode updates the PIN_FLD_DESCR field and any inherited data fields in the session object. Only the fields specified in the opcode's input flist are updated; all other fields in the **/event/session** object are left unchanged. This opcode uses the generalized PIN_FLD_INHERITED_INFO substruct so it can update **/event/session** objects of any type.

Recording the End of a Session

Use the PCM_OP_ACT_LOGOUT opcode to record the end of a login session. This opcode calls the PCM_OP_ACT_END_SESSION opcode to end the session and assess any changes.

Rating and Recording Session Events

Use the PCM_OP_ACT_END_SESSION opcode to rate and record the end of a session event.

PCM_OP_ACT_END_SESSION performs the following operations:

1. Records the session's ending time stamp.
2. Updates the PIN_FLD_DESCR field and any inherited data fields in the **/event/session** object.
3. Returns the following, depending on the flags passed in:
   - When the PCM_OPFLG_READ_RESULT flag is set, the opcode returns all fields in the event object, in addition to the POID.
   - When the PCM_OPFLG_CALC_ONLY flag is set, the opcode returns the fields that would have been used to create the event object. No fields in the database are changed, and the event object is not actually created.
   - When no flags are set, the opcode returns the POID of the event object.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Loading Sessions in Real Time

Use the PCM_OP_ACT_LOAD_SESSION opcode to create, rate, and record a session event as a single operation. This opcode is valuable as a tool to load sessions in real time.

Managing Price Lists

You specify how to charge for the services and products offered by your company by creating a price list. This data is then stored in the BRM database as pricing objects, such as **/deal**, **/plan**, and **/product** objects.

For information on how BRM uses price lists, see "About Creating a Price List".

You can create, retrieve, modify, or delete price list data individually or globally.

- To create, modify, or delete all objects in a price list, use the PCM_OP_PRICE_SET_PRICE_LIST opcode. See "Committing Price List Data to the BRM Database".
- To retrieve all objects in a price list, use the PCM_OP_PRICE_GET_PRICE_LIST opcode. See "Retrieving Price List Data from the BRM Database".
- To create, modify, or delete individual pricing objects, use the PCM_OP_PRICE_COMMIT_* standard opcodes. See "Managing Individual Pricing Objects".

Committing Price List Data to the BRM Database

Use the PCM_OP_PRICE_SET_PRICE_LIST opcode to commit all data for a price list, including plans, products, offer profiles, and deals, to the BRM database. This opcode is called directly by Pricing Center, Customer Center, your custom client application, or the **loadpricelist** utility to load price list data into the database.

> **Note**
>
> Do not use PCM_OP_PRICE_SET_PRICE_LIST to create or update pipeline discount data such as discount models.

PCM_OP_PRICE_SET_PRICE_LIST creates one input flist that contains information for creating, modifying, or deleting the following pricing objects:

- **/best_pricing**
- **/deal**
- **/dependency**
- **/discount**
- **/offer_profile**
- **/plan**
- **/product**
- **/sponsorship**

Copyright © 2025, 2026, Oracle and/or its affiliates.

- **/transition**

PCM_OP_PRICE_SET_PRICE_LIST performs the following in one transaction:

- Verifies the relationships and dependencies between the pricing objects
- Calls the PCM_OP_SUBSCRIPTION_POL_VALIDATE_OFFERING policy opcode. By default, this policy opcode is an empty hook, but you can customize it to validate any product specification attributes defined for pricing objects. See "Defining Product Specification Attributes for Pricing Components" for more information.
- Commits all objects to the BRM database
- Prepares a list of all **/sponsorship** objects that were created or modified and generates the **/event/notification/price/sponsorships/modify** notification event
- Prepares a list of all **/product** objects that were created or modified and generates the **/event/notification/price/products/modify** notification event
- Prepares a list of all **/offer_profile** objects that were created or modified and generates the **/event/notification/price/offer_profile/modify** notification event
- Prepares a list of all **/discount** objects that were created or modified and generates the **/event/notification/price/discounts/modify** notification event

Retrieving Price List Data from the BRM Database

Use the PCM_OP_PRICE_GET_PRICE_LIST opcode to retrieve price list data from the BRM database. This opcode is called directly by the following applications to retrieve price list data:

- Pricing Center
- Customer Center
- A custom client application
- **loadpricelist**, when used with the **-r** parameter

PCM_OP_PRICE_GET_PRICE_LIST retrieves data about the following pricing objects, rebuilds the product flist, and then returns the data back to the calling application:

- **/best_pricing**
- **/deal**
- **/dependency**
- **/discount**
- **/offer_profile**
- **/plan**
- **/product**
- **/sponsorship**
- **/transition**

By default, this opcode returns all price list data. However, you can specify that the opcode retrieve only **/product**, **/discount**, or **/sponsorship** objects by passing optional input flist fields:

- To retrieve only **/product**, **/discount**, or **/sponsorship** objects, pass the PIN_FLD_FLAGS input flist field set to **0x10** for **/product** objects, **0x20** for **/discount** objects, and **0x40** for **/sponsorship** objects. You can specify multiple object types by adding these values; for example, set PIN_FLD_FLAGS to **0x30** to retrieve both **/product** and **/discount** objects.

Copyright © 2025, 2026, Oracle and/or its affiliates.

You can also set the field to **0x0008** to retrieve tailor-made products or other customized pricing data from the BRM system.

- To retrieve **/product** and **/discount** objects based on the service type, pass the PIN_FLD_PERMITTED input flist field set to the desired service type, such as **/service/email**.
- To retrieve **/product**, **/discount**, or **/sponsorship** objects based on the object modification time, pass the PIN_FLD_MOD_T input flist field set to the desired time stamp.

If the opcode retrieves customized pricing data, the output flist includes the PIN_FLD_TAILORMADE field in the product array with a value of **1**. The output flist also includes PIN_FLD_TAILORMADE_DATA, which lists the resources modified by the customization and the percentage changes.

The purchase, cycle, and usage validity periods of products and discounts are returned in the PIN_FLD_PRODUCTS and PIN_FLD_DISCOUNTS arrays in the output flist. If the purchase, cycle, or usage period has a relative start or end time, the relative offset information is returned in respective START_UNIT and START_OFFSET or END_UNIT and END_OFFSET fields. The unit fields specify the type of offset unit, such as minutes, days, or accounting cycles. The offset fields specify the number of offset units in the relative period.

If the purchase, cycle, or usage period starts on first usage (when the customer first uses the product or discount) and the validity period has not yet been set, the START_OFFSET field has a value of **-1**.

The validity period of resources is returned in the PIN_FLD_BAL_IMPACTS array in the PIN_FLD_RATES array. If the resource has a relative start or end time, the relative offset information is returned in the PIN_FLD_RELATIVE_START_UNIT and PIN_FLD_RELATIVE_START_OFFSET or PIN_FLD_RELATIVE_END_UNIT and PIN_FLD_RELATIVE_END_OFFSET fields. For resources, the unit fields specify only cycles. The offset fields specify the number of offset cycles in the relative period.

If the resource starts on first usage (when the customer consumes the resource for the first time) and the validity period has not yet been set, the PIN_FLD_RELATIVE_START_OFFSET field has a value of **-1**.

Managing Individual Pricing Objects

You can manage an individual pricing object, such as a **/deal** object, by using the PCM_OP_PRICE_COMMIT_* standard opcodes. These opcodes create, modify, or delete objects by using data in the input flist only; they do not verify relationships or dependencies with other pricing objects before committing changes to the database. For example, the PCM_OP_PRICE_COMMIT_PRODUCT opcode creates a **/product** object without first verifying that it is associated with a deal in the price list.

> **Note**
>
> Because these opcodes do not verify dependencies with other pricing objects, you should use them in development environments only. To create pricing objects in a production environment, use the PCM_OP_PRICE_SET_PRICE_LIST opcode. See "Committing Price List Data to the BRM Database".

You use the following opcodes to manage individual pricing objects:

- PCM_OP_PRICE_COMMIT_PRODUCT. See "Managing /product Objects".

Copyright © 2025, 2026, Oracle and/or its affiliates.

- PCM_OP_PRICE_COMMIT_DISCOUNT. See "Managing /discount Objects".
- PCM_OP_PRICE_COMMIT_DEAL. See "Managing /deal Objects".
- PCM_OP_PRICE_COMMIT_PLAN. See "Managing /plan Objects".
- PCM_OP_PRICE_COMMIT_DEPENDENCY. See "Managing /dependency Objects".
- PCM_OP_PRICE_COMMIT_TRANSITION. See "Managing /transition Objects".
- PCM_OP_PRICE_COMMIT_PLAN_LIST. See "Managing /group/plan_list Objects".
- PCM_OP_PRICE_COMMIT_SPONSORSHIP. See "Managing /sponsorship Objects".

> **Note**
>
> Each of these standard opcodes calls PREP and VALID policy opcodes before committing changes to the database. You can use these policy opcodes to add data to the objects or to perform custom validation.

Managing /product Objects

Use the PCM_OP_PRICE_COMMIT_PRODUCT opcode to create, modify, or delete **/product** objects in the BRM database.

To create or modify **/product** objects, pass details about the product in the opcode's PIN_FLD_PRODUCT input flist array. In the array, you must also pass in the PIN_FLD_CODE field set to the **/product** object's unique identifier and the PIN_FLD_NAME field set to the product's name, which is modifiable.

This opcode validates and commits the following objects from a product flist: **/rate**, **/rate_plan**, **/rate_plan_selector**, and **/rollover**. Products are created or modified and, if the delete flag is sent in, deleted.

/rate_plan objects can have multiple rates, one for each rate tier (priority), date, day, and time-of-day range. These ranges are structured into a four-level tree of rate tier, date, day, and time. An optional rate plan selector is used when multiple rate plans from which to select are available.

When a product grants a noncurrency resource, such as free minutes, the validity period of the granted resource is stored in the **/rate** object's PIN_FLD_BAL_IMPACTS array. See "Managing the Validity Period of Granted Resources" for more information.

PCM_OP_PRICE_COMMIT_PRODUCT performs the following in one transaction:

- Generates the **/event/notification/price/products/modify** notification event
- Calls the PCM_OP_SUBSCRIPTION_POL_VALIDATE_OFFERING policy opcode. By default, this policy opcode is an empty hook, but you can customize it to validate any product specification attributes defined for **/product** objects. See "Defining Product Specification Attributes for Pricing Components" for more information.
- Commits all objects to the BRM database
- Returns the following in the output flist:
  - If all operations are successful, this opcode returns PIN_FLD_RESULTS set to **1** and PIN_FLD_PRODUCTS set to the **/product** POID.

Copyright © 2025, 2026, Oracle and/or its affiliates.

  - If any operation fails, this opcode returns PIN_FLD_RESULTS set to **0**. It also returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

Managing the Validity Period of Granted Resources

Resource validity periods define when a granted resource is available for consumption.

To set a resource's start and end times, pass the following values in the PIN_FLD_BAL_IMPACTS array in the PIN_FLD_RATES array of the PCM_OP_PRICE_COMMIT_PRODUCT input flist:

- Set the PIN_FLD_FLAGS field to PIN_RATE_BAL_FLG_GRANTABLE (**0x08**). This specifies that the balance impact is granting the resource.
- Specify the start times as follows:
  - To start on a specific date, pass the date in PIN_FLD_START_T.
  - To start immediately, set the value of PIN_FLD_START_T to **0**.
  - To start on first usage (when the customer impacts the resource balance for the first time), set the value of PIN_FLD_RELATIVE_START_OFFSET to **-1**.
  - To start relative to the grant date, set the value of PIN_FLD_RELATIVE_START_UNIT to the type of relative unit and set the number of units in PIN_FLD_RELATIVE_START_OFFSET.
- Specify the end time as follows:
  - To never end, set the value of PIN_FLD_END_T to **0**.
  - To end relative to the start date, set the value of PIN_FLD_RELATIVE_END_UNIT to the type of relative unit and set the number of units in PIN_FLD_RELATIVE_END_OFFSET.

For more information about how to specify the unit and offset values, see "Specifying Relative Start and End Times for Products and Discounts in a Deal".

When the deal is committed to the BRM database, any relative validity information is stored in the PIN_FLD_START_DETAILS and PIN_FLD_END_DETAILS fields in the **/rate** object's PIN_FLD_BAL_IMPACTS array.

When the resource is granted to an account, any relative validity information is stored in the PIN_FLD_VALID_FROM_DETAILS and PIN_FLD_VALID_TO_DETAILS fields in the account **/balance_group** object's PIN_FLD_BAL_IMPACTS array.

For information about the values stored in details fields, see "Storing Start and End Times for Products and Discounts in a Deal".

Specifying Relative Start and End Times for Products and Discounts in a Deal

When the purchase, cycle, or usage period of a product or discount has a relative start or end time, the details about the relative period are passed in opcode flists in UNIT and OFFSET fields.

The following UNIT fields specify the type of relative unit:

- PIN_FLD_PURCHASE_START_UNIT
- PIN_FLD_PURCHASE_END_UNIT
- PIN_FLD_CYCLE_START_UNIT

Copyright © 2025, 2026, Oracle and/or its affiliates.

- PIN_FLD_CYCLE_END_UNIT
- PIN_FLD_USAGE_START_UNIT
- PIN_FLD_USAGE_END_UNIT

The unit can be one of these values:

- Seconds = 1
- Minutes = 2
- Hours = 3
- Days = 4
- Months = 5
- Accounting cycles = 8

The following OFFSET fields specify the number of units in the relative period:

- PIN_FLD_PURCHASE_START_OFFSET
- PIN_FLD_PURCHASE_END_OFFSET
- PIN_FLD_CYCLE_START_OFFSET
- PIN_FLD_CYCLE_END_OFFSET
- PIN_FLD_USAGE_START_OFFSET
- PIN_FLD_USAGE_END_OFFSET

The UNIT and OFFSET fields for each purchase, cycle, and usage period are used in combination to determine the relative offset period. For example:

- If a product's PIN_FLD_CYCLE_START_UNIT has a value of **8** (accounting cycles) and PIN_FLD_CYCLE_START_OFFSET has a value of **3**, the cycle fee starts three accounting cycles after the product is purchased.
- If a discount's PIN_FLD_PURCHASE_END_UNIT has a value of **8** (accounting cycles) and PIN_FLD_PURCHASE_END_OFFSET has a value of **3**, the discount expires three accounting cycles after it is activated.

You can specify any number of units in the OFFSET fields, up to 1048576. This is equivalent to approximately 12 days in seconds, 728 days in minutes, and 119 years in hours.

Information about the relative offset validity periods is stored in the BRM database in DETAILS fields in the **/deal** object's products and discounts arrays. For more information, see "Storing Start and End Times for Products and Discounts in a Deal".

The product and discount start and end dates (the PIN_FLD_*_START_T and PIN_FLD_*_END_T fields) are not set in the **/deal** object but are set in the **/purchased_product** and **/purchased_discount** objects when the product or discount is purchased.

Storing Start and End Times for Products and Discounts in a Deal

When a product or discount has a relative start or end time, the relative information passed in the PIN_FLD_*_UNIT and PIN_FLD_*_OFFSET fields of opcode flists is stored in the database in PIN_FLD_*_DETAILS fields in the **/deal** object:

- PIN_FLD_PURCHASE_START_DETAILS
- PIN_FLD_PURCHASE_END_DETAILS
- PIN_FLD_CYCLE_START_DETAILS

Copyright © 2025, 2026, Oracle and/or its affiliates.

- PIN_FLD_CYCLE_END_DETAILS
- PIN_FLD_USAGE_START_DETAILS
- PIN_FLD_USAGE_END_DETAILS

The START-DETAILS and END-DETAILS fields are 32-bit integer fields that store three values: the mode of the validity period, the relative offset unit, and the number of offset units in the relative period:

- **Mode**: The mode specifies generally when the validity period starts or ends. The mode also helps to determine how the start and end dates are set in the account's **/purchased_product** and **/purchased_discount** objects when the product or discount is purchased. The mode is stored in the lower 8th bit and can have one of the following values:

  For the START-DETAILS:

  - **1** = PIN_VALIDITY_IMMEDIATE: The validity period starts when the product or discount is purchased. When purchased, the PIN_FLD_*_START_T field in the account's **/purchased_product** or **/purchased_discount** object is set to the purchase time.
  - **3** = PIN_VALIDITY_FIRST_USAGE: The validity period starts when the product or discount is first used to rate the customer's usage.
  - **4** = PIN_VALIDITY_RELATIVE: The validity period starts relative to the purchase date. When the product or discount is purchased, the start date is calculated by adding the relative offset period to the purchase date. The start date is then set in the purchased product's or purchased discount's PIN_FLD_*_START_T field.

  For the END-DETAILS:

  - **2** = PIN_VALIDITY_NEVER: The validity period never ends. When the product or discount is purchased, the purchased product's or purchased discount's PIN_FLD_*_END_T field is set to **0**.
  - **4** = PIN_VALIDITY_RELATIVE: The validity period ends relative to when it starts. When the product or discount is purchased, the end date is calculated by adding the relative offset period to the start date. The end date is then set in the purchased product's or purchased discount's PIN_FLD_*_END_T field.

  When PIN_FLD_*_START_T and PIN_FLD_*_END_T are set in the account's **/purchased_product** and **/purchased_discount** objects, the mode in the details fields of those objects is set to PIN_VALIDITY_ABSOLUTE.

- **Relative offset unit**: This value is set when the product or discount starts at a time relative to the purchase date or ends at a time relative to the start date. It specifies the type of offset unit and corresponds to the value of PIN_FLD_*_START_UNIT or PIN_FLD_*_END_UNIT that is passed in opcode flists. The relative offset unit is stored in the next four bits of the details field and can have one of the following values:

  - Seconds = 1
  - Minutes = 2
  - Hours = 3
  - Days = 4
  - Months = 5
  - Accounting cycles = 8

- **Number of offset units**: This value specifies how many offset units are in the relative period and corresponds to the value of PIN_FLD_*_START_OFFEST or

Copyright © 2025, 2026, Oracle and/or its affiliates.

PIN_FLD_*_END_OFFSET that is passed in opcode flists. The number of offset units is stored in the remaining 20 bits of the details field.

Modifying Rate Types

Use the PCM_OP_PRICE_SET_PRICE_LIST opcode to modify the type of a rate in a **/product** object. The rate type controls whether customers can exceed their credit limit.

To modify the **PIN_FLD_TYPE** field in **/rate** objects, pass details about the product and rate in the opcode's PIN_FLD_PRODUCT input flist array.

You can set the type to any of the following:

- **0**: The credit limit is enforced. The subscription succeeds and the resources are prorated according to the available balance.
- **740**: The credit limit is exceeded. The subscription succeeds and all available balance is used. The remaining amount is recorded as an outstanding amount to be paid at the next top up for prepaid customers or current bill for postpaid customers.
- **742**: The credit limit is exceeded. The subscription succeeds. If the customer is eligible for a loan, all available balance is used and a loan is granted for the remaining amount. If the customer is not eligible for a loan, the subscription fails.
- **743**: The credit limit is exceeded. The subscription succeeds and all available balance is used. The remaining amount is recorded as an outstanding amount to be paid at the next top up for prepaid customers or current bill for postpaid customers. This option is intended for prepaid customers.
- **744**: The credit limit is exceeded. The subscription succeeds without using the available balance. The entire amount is recorded as an outstanding amount to be paid at the next top up for prepaid customers or current bill for postpaid customers.
- **745**: The credit limit is enforced. The subscription fails and after a configured maximum number of retries, a notification event is sent to an external system for further processing.
- **746**: The credit limit is exceeded. The subscription succeeds without using the available balance. Billing will be skipped for this cycle; during rating all balance impacts are dropped and the unrated quantity is set to 0. The cycle forward date is moved to next cycle.

For example, to set the rate type to 746 for a product, you would use the following input flist:

```
0 PIN_FLD_PRODUCTS      ARRAY [0] allocated 20, used 20
1     PIN_FLD_RATE_PLANS    ARRAY [0] allocated 20, used 9
2         PIN_FLD_TAX_WHEN       ENUM [0] 0
2         PIN_FLD_CYCLE_FEE_FLAGS    INT [0] 0
2         PIN_FLD_CURRENCY        INT [0] 840
2         PIN_FLD_NAME            STR [0] "cycle_forward_monthly"
2         PIN_FLD_RATES         ARRAY [0] allocated 20, used 7
3             PIN_FLD_STEP_RESOURCE_ID    INT [0] 840
3             PIN_FLD_PRORATE_LAST   ENUM [0] 701
3             PIN_FLD_PRORATE_FIRST   ENUM [0] 702
3             PIN_FLD_TYPE           ENUM [0] 746
```

Customizing How to Create and Delete Products

To customize how to create and delete products:

- To enable data modification during **/product** object creation, use the PCM_OP_PRICE_POL_PREP_PRODUCT policy opcode. Use this policy opcode to

Copyright © 2025, 2026, Oracle and/or its affiliates.

enhance **/product** objects with additional data not provided by either the GUI application or by the Price List FM.

The PCM_OP_PRICE_COMMIT_PRODUCT opcode calls this policy opcode before creating a new **/product** object.

> **Note**
>
> This policy opcode is called before the Price List FM performs any validation checks.

- To enable validation during **/product** object creation, use the PCM_OP_PRICE_POL_VALID_PRODUCT policy opcode. This policy opcode can be used to perform validations in addition to those performed by the Price List FM.

  The PCM_OP_PRICE_COMMIT_PRODUCT opcode calls this policy opcode before creating a **/product** object. By default, this policy opcode is an empty hook that returns the result PIN_PRICE_VERIFY_PASSED.

- To verify that deleting a **/product** object is permitted, use the PCM_OP_PRICE_POL_DELETE_PRODUCT policy opcode. Use this policy opcode to perform validations in addition to those performed by the Price List FM.

  PCM_OP_PRICE_COMMIT_PRODUCT calls this policy opcode before deleting the **/product** object. By default, it returns the result PIN_PRICE_VERIFY_PASSED.

  Success or failure is indicated by the value returned in the PIN_FLD_RESULTS output flist field: a value of **1** indicates that the operation succeeded and a value of **0** indicates that the operation failed.

Using Opcodes to Customize Products

You can customize products by changing individual rates and price models for specific resources. Customer service representatives (CSRs) can use this capability in Customer Center, but you can also call opcodes from third-party CRM applications.

> **Note**
>
> Customer Center enforces permissioning requirements for product customization. Permissioning is not enforced when you customize products by calling opcodes directly. You must implement permissioning in the CRM tool that calls the opcodes.

Creating a Customized Product

To create custom products:

- Call the PCM_OP_PRICE_GET_PRODUCT_INFO opcode to retrieve information about the base product, based on its POID.
- Based on information about the base product and on the customization information (resources modified and the percentage modification), use the PCM_OP_PRICE_PREP_TAILORMADE_PRODUCT opcode to prepare an flist for the new customized **/product** object.
- Call the PCM_OP_SUBSCRIPTION_PURCHASE_DEAL opcode to purchase the customized product and create a **/purchased_product** object to represent it in the

Copyright © 2025, 2026, Oracle and/or its affiliates.

account. This opcode calls the PCM_OP_SUBSCRIPTION_PURCHASE_PRODUCT opcode.

- Pass the flist prepared by PCM_OP_PRICE_PREP_TAILORMADE_PRODUCT to the PCM_OP_PRICE_SET_PRICE_LIST opcode to commit the customized **/product** object to the BRM database. See "Committing Price List Data to the BRM Database".
- Call the PCM_OP_SUBSCRIPTION_CYCLE_FORWARD opcode to apply the appropriate cycle fees. See "About Applying Cycle Forward Fees".

Modifying a Customized Product

From Customer Center, the only change you can make to an existing customized product is to modify its validity dates. This limitation does not apply when modifying customized products by calling opcodes directly, however.

- Specify changes to the customized product's rates and price models and pass them to PCM_OP_PRICE_PREP_TAILORMADE_PRODUCT to prepare an flist for the modified customized **/product** object.
- Pass the flist to PCM_OP_PRICE_SET_PRICE_LIST to update the **/product** object. See "Committing Price List Data to the BRM Database".
- Call the PCM_OP_SUBSCRIPTION_SET_PRODINFO opcode to update the **/purchased_product** object associated with the customized product.

Setting Full Day Proration

Use the PCM_OP_PRICE_SET_PRICE_LIST opcode to configure full day proration in the **/product** object by using the PIN_FLD_OFFER_VALIDITY_ROUNDING and PIN_FLD_SCALE_ROUNDING fields.

PIN_FLD_OFFER_VALIDITY_ROUNDING specifies whether to start the charge offer's validity period at the purchase time or midnight of the purchase day.

Set the PIN_FLD_OFFER_VALIDITY_ROUNDING field to one of the following:

- **0**: Uses your company’s systemwide setting.
- **1**: Starts at midnight (00:00:00) of the day that the charge offer is purchased.
- **2**: Starts at the time of purchase.

PIN_FLD_SCALE_ROUNDING specifies how to calculate the scale.

Set the PIN_FLD_SCALE_ROUNDING field to one of the following:

- **0**: Calculate it based on the PIN_FLD_SCALE_ROUNDING setting.
- **1**: Calculate it based on full days.

Managing /discount Objects

Use the PCM_OP_PRICE_COMMIT_DISCOUNT opcode to:

- Create, modify, or delete **/discount** objects in the BRM database
- Create pipeline discount models

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> The opcode cannot modify pipeline discount models.

Creating /discount Objects

To create **/discount** objects, pass details about the discount in the opcode's PIN_FLD_DISCOUNT input flist array. You can also optionally create a pipeline discount model by using the PIN_FLD_PIPELINE_DISC_MODELS input flist array. See "Creating Pipeline Discount Models" for more information.

An offer profile and the provisioning tag for the associated discount use the same name and that name must be unique. If you create an offer profile to associate with existing discount, use the provisioning tag in the discount object to name the offer profile. When you configure a new discount around an existing offer profiles, use the appropriate offer profile name as the provisioning tag for the discount. Use the resource specified in the offer profile as a tracking resource in the discount that is attached to the offer profile.

When the **/discount** object does not exist in the database, PCM_OP_PRICE_COMMIT_DISCOUNT creates a **/discount** object by doing the following:

- Calling the PCM_OP_PRICE_POL_PREP_DISCOUNT policy opcode to perform any custom preparation. By default, this policy opcode is an empty hook, but you can customize it to enhance **/discount** objects with additional data or to perform other custom actions. See "Customizing /discount Objects" for more information.
- Calling the PCM_OP_PRICE_POL_VALID_DISCOUNT policy opcode to perform any custom validation. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions. See "Customizing /discount Objects" for more information.
- Calling the PCM_OP_SUBSCRIPTION_POL_VALIDATE_OFFERING policy opcode. By default, this policy opcode is an empty hook, but you can customize it to validate any product specification attributes defined for **/discount** objects. See "Defining Product Specification Attributes for Pricing Components" for more information.
- Creating the specified **/discount** object in the BRM database.
- Generating the **/event/notification/price/discounts/modify** notification event. This enables you to synchronize discounts between BRM and external CRM applications.

PCM_OP_PRICE_COMMIT_DISCOUNT performs all operations within a single transaction, so success or failure is indicated by the value returned for PIN_FLD_RESULTS:

- When successful, PCM_OP_PRICE_COMMIT_DISCOUNT sets PIN_FLD_RESULTS to **1**.
- When any operation fails, PCM_OP_PRICE_COMMIT_DISCOUNT sets PIN_FLD_RESULTS to **0**. It also returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

Creating Pipeline Discount Models

To create pipeline discount models, pass details about the discount model in the PCM_OP_PRICE_COMMIT_DISCOUNT opcode's PIN_FLD_PIPELINE_DISC_MODELS input flist array. The array can contain the following information:

- Discount model version and configuration

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Discount/chargeshare trigger
- Discount/chargeshare condition
- Discount/chargeshare rules
- Discount/chargeshare master
- Discount/chargeshare detail
- Discount/chargeshare step

The discount model, trigger, rule, step, and master are uniquely identified by a code string in the PIN_FLD_CODE_STR field. If a code string is not passed in the input flist, the opcode generates one automatically.

Modifying /discount Objects

To modify **/discount** objects, pass details about the discount in the opcode's PIN_FLD_DISCOUNT input flist array. In the array, you must also pass in the PIN_FLD_CODE field set to the **/discount** object's unique identifier and the PIN_FLD_NAME field set to the discount's name, which is modifiable.

> **Note**
>
> This opcode overwrites data in existing **/discount** objects, so be sure you pass in the correct object to modify.

When the object already exists in the database, PCM_OP_PRICE_COMMIT_DISCOUNT modifies the **/discount** object by doing the following:

- Overwriting the specified **/discount** object in the BRM database.
- Generating the **/event/notification/price/discounts/modify** notification event. This enables you to synchronize discounts between BRM and external CRM applications.

PCM_OP_PRICE_COMMIT_DISCOUNT performs all operations within a single transaction, so success or failure is indicated by the value returned for PIN_FLD_RESULTS:

- When successful, PCM_OP_PRICE_COMMIT_DISCOUNT sets PIN_FLD_RESULTS to **1**.
- When any operation fails, PCM_OP_PRICE_COMMIT_DISCOUNT sets PIN_FLD_RESULTS to **0**. It also returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

Deleting /discount Objects

To delete **/discount** objects, pass details about the discount in the opcode's PIN_FLD_DISCOUNT input flist array and set the PIN_FLD_DELETED_FLAG input flist field to **1**.

When PIN_FLD_DELETED_FLAG is set, PCM_OP_PRICE_COMMIT_DISCOUNT deletes the **/discount** object by doing the following:

- Calling the PCM_OP_PRICE_POL_DELETE_DISCOUNT policy opcode to perform any custom validation. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.
- Deleting the specified **/discount** object in the BRM database.

Copyright © 2025, 2026, Oracle and/or its affiliates.

PCM_OP_PRICE_COMMIT_DISCOUNT performs all operations within a single transaction, so success or failure is indicated by the value returned for PIN_FLD_RESULTS:

- When successful, PCM_OP_PRICE_COMMIT_DISCOUNT sets PIN_FLD_RESULTS to **1**.
- When any operation fails, PCM_OP_PRICE_COMMIT_DISCOUNT sets PIN_FLD_RESULTS to **0**. It also returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

Customizing /discount Objects

You can customize the **/discount** object before it is committed to the database by using the Price List FM policy opcodes:

- To enable data modification during **/discount** object creation, use the PCM_OP_PRICE_POL_PREP_DISCOUNT policy opcode. Use this policy opcode to enhance **/discount** objects with additional data not provided by either the GUI application or by the Price List FM.

  This policy opcode is called by PCM_OP_PRICE_COMMIT_DISCOUNT. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.

  Success or failure is indicated by the value returned in the PIN_FLD_RESULTS output flist field: a value of **1** indicates that all operations succeeded and a value of **0** indicates that the operation failed.

- To enable validation during **/discount** object creation, use the PCM_OP_PRICE_POL_VALID_DISCOUNT policy opcode. This policy opcode can be used to enhance **/discount** objects with additional data not provided by either Pricing Center or by other opcodes in the Price List FM.

  This policy opcode is called by PCM_OP_PRICE_COMMIT_DISCOUNT before creating a **/discount** object. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.

  This policy opcode is called before the Price List FM performs any default validation checks.

  Success or failure is indicated by the value returned in the PIN_FLD_RESULTS output flist field: a value of **1** indicates that all operations succeeded and a value of **0** indicates that the operation failed.

- To verify that deleting a **/discount** object is permitted, use the PCM_OP_PRICE_POL_DELETE_DISCOUNT policy opcode. Use this policy opcode to perform validations in addition to those performed by the Price List FM.

  This policy opcode is called by PCM_OP_PRICE_COMMIT_DISCOUNT before it deletes a **/discount** object. By default, this policy opcode is an empty hook, but you can customize it to enhance **/discount** objects with additional data or to perform other customizations.

  This policy opcode is intended as a place for you to add validation checks or other functionality before allowing a **/discount** object to be deleted. For example, you may want to confirm that the valid time period for the **/discount** object has expired before allowing it to be deleted.

  Success or failure is indicated by the value returned in the PIN_FLD_RESULTS output flist field: a value of **1** indicates that all operations succeeded and a value of **0** indicates that the operation failed.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Configuring Discount Validity Rounding

Use the PCM_OP_PRICE_SET_PRICE_LIST opcode to configure discount validity rounding in the /discount object by using the PIN_FLD_DISCOUNT_VALIDITY_ROUNDING field.

PIN_FLD_DISCOUNT_VALIDITY_ROUNDING field specifies whether to start the discount offer's validity period at the purchase time or midnight of the purchase day.

Set the PIN_FLD_DISCOUNT_VALIDITY_ROUNDING field to one of the following:

- 0: Uses your company's systemwide setting. This is the default.
- 1: Starts at midnight (00:00:00) of the day that the discount offer is purchased.
- 2: Starts at the time of purchase.

Retrieving Discount Data

Use the PCM_OP_PRICE_GET_DISCOUNT_INFO opcode to retrieve details about the real-time **/discount** object along with the pipeline discount model information from the BRM database. The discount model data includes the following:

- Discount model version and configuration
- Discount/chargeshare trigger
- Discount/chargeshare condition
- Discount/chargeshare rules
- Discount/chargeshare master
- Discount/chargeshare detail
- Discount/chargeshare step
- Balance impact

This opcode takes as input the discount object POID and optionally the discount object name. This opcode performs the following:

1. Retrieves the real-time **/discount** object details from the BRM database.
2. Retrieves the related pipeline discount model information used by the **/discount** object.
3. Returns details about the real-time **/discount** object and pipeline discount model in the output flist.

Managing /deal Objects

Use the PCM_OP_PRICE_COMMIT_DEAL opcode to create, change, or delete **/deal** objects in the BRM database.

This opcode accepts an flist consisting of a PIN_FLD_DEAL array, each element of which represents a **/deal** object.

PCM_OP_PRICE_COMMIT_DEAL determines whether to create, modify, or delete the specified **/deal** object:

- When the object does not exist, this opcode *creates* the specified **/deal** object by doing the following:

Copyright © 2025, 2026, Oracle and/or its affiliates.

  - Calling the PCM_OP_PRICE_POL_PREP_DEAL policy opcode to perform any custom preparation. See "Customizing How to Create and Delete Deals" for more information.
  - Calling the PCM_OP_PRICE_POL_VALID_DEAL policy opcode to perform any custom validation. See "Customizing How to Create and Delete Deals" for more information.
  - Calling the PCM_OP_SUBSCRIPTION_POL_VALIDATE_OFFERING policy opcode. By default, this policy opcode is an empty hook, but you can customize it to validate any product specification attributes defined for **/deal** objects. See "Defining Product Specification Attributes for Pricing Components" for more information.
  - Creating the specified object in the BRM database.
- When the object already exists, this opcode *modifies* the specified **/deal** object in the BRM database.

> **Note**
>
> PCM_OP_PRICE_COMMIT_DEAL overwrites data in existing **/deal** objects, so be sure you pass in the correct object to modify.

- When the PIN_FLD_DELETED_FLAG is set, this opcode *deletes* the specified **/deal** object by doing the following:
  - Calling the PCM_OP_PRICE_POL_DELETE_DISCOUNT policy opcode to perform any custom validation. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.
  - Deleting the specified object in the BRM database.

PCM_OP_PRICE_COMMIT_DEAL performs all operations within a single transaction, so success or failure is indicated by the value returned in the PIN_FLD_RESULTS output flist field: a value of **1** indicates that all operations succeeded and a value of **0** indicates that the operation failed. When the operation fails, the opcode also returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

To set the purchase, cycle, and usage start and end dates of the products and discounts in the deal, pass the following values in opcode input flists in the PIN_FLD_PRODUCTS and PIN_FLD_DISCOUNTS arrays. In the following fields, the asterisk (*) indicates either PURCHASE, CYCLE, or USAGE.

> **Note**
>
> The cycle and usage validity periods must fall within the purchase validity period.

- To start immediately, set the value of PIN_FLD_*_START_OFFSET to **0**.
- To start on first usage (when the customer uses the product or discount for the first time), set the value of PIN_FLD_*_START_OFFSET to **-1**.
- To start relative to the purchase date, set the relative unit (for example, days or cycles) in PIN_FLD_*_START_UNIT and set the number of relative units in PIN_FLD_*_START_OFFSET.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> If you specify a relative unit for more than one period (purchase, cycle, or usage) and you select a different relative unit for each, if one of the relative units is cycles, the PCM_OP_PRICE_POL_VALID_DEAL policy opcode cannot validate that the cycle or usage period falls within the purchase period. The validation for this kind of configuration is performed when the product is purchased by a customer.

- To never end, set the value of PIN_FLD_*_END_OFFSET to **0**.
- To end relative to the start date, set the relative unit (for example, days or cycles) in PIN_FLD_*_END_UNIT and set the number of relative units in PIN_FLD_*_END_OFFSET.

For more information about how to specify the unit and offset values, see "Specifying Relative Start and End Times for Products and Discounts in a Deal" for more information.

If the deal's products or discounts grant resources that start on first usage (when the customer impacts the resource balance for the first time), you can specify that the validity period of all resources in the deal that start on first usage are set when one of those resources is impacted for the first time. To do this, set the PIN_FLD_GRANT_RESOURCES_AS_GROUP flag value in the PIN_FLD_FLAGS field.

When the deal is committed to the BRM database, the purchase, cycle, and usage validity information is stored in the PIN_FLD_*_START_DETAILS and PIN_FLD_*_END_DETAILS fields in the **/deal** object's product and discount arrays. For information about the values stored in the details fields, see "Storing Start and End Times for Products and Discounts in a Deal" for more information.

Specifying Relative Start and End Times for Products and Discounts in a Deal

When the purchase, cycle, or usage period of a product or discount has a relative start or end time, the details about the relative period are passed in opcode flists in UNIT and OFFSET fields.

The following UNIT fields specify the type of relative unit:

- PIN_FLD_PURCHASE_START_UNIT
- PIN_FLD_PURCHASE_END_UNIT
- PIN_FLD_CYCLE_START_UNIT
- PIN_FLD_CYCLE_END_UNIT
- PIN_FLD_USAGE_START_UNIT
- PIN_FLD_USAGE_END_UNIT

The unit can be one of these values:

- Seconds = 1
- Minutes = 2
- Hours = 3
- Days = 4
- Months = 5
- Accounting cycles = 8

The following OFFSET fields specify the number of units in the relative period:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- PIN_FLD_PURCHASE_START_OFFSET
- PIN_FLD_PURCHASE_END_OFFSET
- PIN_FLD_CYCLE_START_OFFSET
- PIN_FLD_CYCLE_END_OFFSET
- PIN_FLD_USAGE_START_OFFSET
- PIN_FLD_USAGE_END_OFFSET

The UNIT and OFFSET fields for each purchase, cycle, and usage period are used in combination to determine the relative offset period. For example:

- If a product's PIN_FLD_CYCLE_START_UNIT has a value of **8** (accounting cycles) and PIN_FLD_CYCLE_START_OFFSET has a value of **3**, the cycle fee starts three accounting cycles after the product is purchased.
- If a discount's PIN_FLD_PURCHASE_END_UNIT has a value of **8** (accounting cycles) and PIN_FLD_PURCHASE_END_OFFSET has a value of **3**, the discount expires three accounting cycles after it is activated.

You can specify any number of units in the OFFSET fields, up to 1048576. This is equivalent to approximately 12 days in seconds, 728 days in minutes, and 119 years in hours.

Information about the relative offset validity periods is stored in the BRM database in DETAILS fields in the **/deal** object's products and discounts arrays. For more information, see "Storing Start and End Times for Products and Discounts in a Deal".

The product and discount start and end dates (the PIN_FLD_*_START_T and PIN_FLD_*_END_T fields) are not set in the **/deal** object but are set in the **/purchased_product** and **/purchased_discount** objects when the product or discount is purchased.

Storing Start and End Times for Products and Discounts in a Deal

When a product or discount has a relative start or end time, the relative information passed in the PIN_FLD_*_UNIT and PIN_FLD_*_OFFSET fields of opcode flists is stored in the database in PIN_FLD_*_DETAILS fields in the **/deal** object:

- PIN_FLD_PURCHASE_START_DETAILS
- PIN_FLD_PURCHASE_END_DETAILS
- PIN_FLD_CYCLE_START_DETAILS
- PIN_FLD_CYCLE_END_DETAILS
- PIN_FLD_USAGE_START_DETAILS
- PIN_FLD_USAGE_END_DETAILS

The START-DETAILS and END-DETAILS fields are 32-bit integer fields that store three values: the mode of the validity period, the relative offset unit, and the number of offset units in the relative period:

- **Mode**: The mode specifies generally when the validity period starts or ends. The mode also helps to determine how the start and end dates are set in the account's **/purchased_product** and **/purchased_discount** objects when the product or discount is purchased. The mode is stored in the lower 8th bit and can have one of the following values:

  For the START-DETAILS:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- **1** = PIN_VALIDITY_IMMEDIATE: The validity period starts when the product or discount is purchased. When purchased, the PIN_FLD_*_START_T field in the account's **/purchased_product** or **/purchased_discount** object is set to the purchase time.
- **3** = PIN_VALIDITY_FIRST_USAGE: The validity period starts when the product or discount is first used to rate the customer's usage.
- **4** = PIN_VALIDITY_RELATIVE: The validity period starts relative to the purchase date. When the product or discount is purchased, the start date is calculated by adding the relative offset period to the purchase date. The start date is then set in the purchased product's or purchased discount's PIN_FLD_*_START_T field.

For the END-DETAILS:

- **2** = PIN_VALIDITY_NEVER: The validity period never ends. When the product or discount is purchased, the purchased product's or purchased discount's PIN_FLD_*_END_T field is set to **0**.
- **4** = PIN_VALIDITY_RELATIVE: The validity period ends relative to when it starts. When the product or discount is purchased, the end date is calculated by adding the relative offset period to the start date. The end date is then set in the purchased product's or purchased discount's PIN_FLD_*_END_T field.

When PIN_FLD_*_START_T and PIN_FLD_*_END_T are set in the account's **/purchased_product** and **/purchased_discount** objects, the mode in the details fields of those objects is set to PIN_VALIDITY_ABSOLUTE.

- **Relative offset unit**: This value is set when the product or discount starts at a time relative to the purchase date or ends at a time relative to the start date. It specifies the type of offset unit and corresponds to the value of PIN_FLD_*_START_UNIT or PIN_FLD_*_END_UNIT that is passed in opcode flists. The relative offset unit is stored in the next four bits of the details field and can have one of the following values:
  - Seconds = 1
  - Minutes = 2
  - Hours = 3
  - Days = 4
  - Months = 5
  - Accounting cycles = 8
- **Number of offset units**: This value specifies how many offset units are in the relative period and corresponds to the value of PIN_FLD_*_START_OFFEST or PIN_FLD_*_END_OFFSET that is passed in opcode flists. The number of offset units is stored in the remaining 20 bits of the details field.

Customizing How to Create and Delete Deals

To customize how to create and delete deals, use the following policy opcodes:

- To enable data modification during **/deal** object creation, use the PCM_OP_PRICE_POL_PREP_DEAL policy opcode.

  By default, this policy opcode is an empty hook, but you can customize it to enhance **/deal** objects with additional data not provided either by the GUI application or by the Price List FM or to perform other custom actions.

  This policy opcode is called before the Price List FM performs any validation checks.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- To enable validation during **/deal** object creation, use the PCM_OP_PRICE_POL_VALID_DEAL policy opcode. This policy opcode can be used to perform validations in addition to those performed by the Price List FM.

  This policy opcode is called by PCM_OP_PRICE_COMMIT_DEAL. By default, this policy opcode is an empty hook; it returns the result PIN_PRICE_VERIFY_PASSED.

- To verify that deleting a **/deal** object is permitted, use the PCM_OP_PRICE_POL_DELETE_DEAL policy opcode. Use this opcode to perform validations in addition to those performed by the Price List FM.

  PCM_OP_PRICE_COMMIT_DEAL calls this policy opcode before deleting the **/deal** object. By default, this policy opcode is an empty hook provided to facilitate customization. It returns the result PIN_PRICE_VERIFY_PASSED.

  Success or failure is indicated by the value returned for PIN_FLD_RESULTS. When successful, PIN_FLD_RESULTS is set to **1**. When any operation fails, PIN_FLD_RESULTS is set to **0**.

Managing /plan Objects

Use the PCM_OP_PRICE_COMMIT_PLAN opcode to create, modify, or delete **/plan** objects in the BRM database.

This opcode accepts an flist consisting of a PIN_FLD_PLAN array, each element of which represents a **/plan** object.

PCM_OP_PRICE_COMMIT_PLAN determines whether to create, modify, or delete the specified **/plan** object:

- When the object does not exist, this opcode *creates* the specified **/plan** object by doing the following:
  - Calling the PCM_OP_PRICE_POL_PREP_DISCOUNT policy opcode to perform any custom preparation. By default, this policy opcode is an empty hook, but you can customize it to enhance **/plan** objects with additional data or to perform other custom actions.
  - Calling the PCM_OP_PRICE_POL_VALID_DISCOUNT policy opcode to perform any custom validation. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.
  - Calling the PCM_OP_SUBSCRIPTION_POL_VALIDATE_OFFERING policy opcode. By default, this policy opcode is an empty hook, but you can customize it to validate any product specification attributes defined for **/plan** objects. See "Defining Product Specification Attributes for Pricing Components" for more information.
  - Creating the specified object in the BRM database.
- When the object already exists, this opcode *modifies* the specified **/plan** object in the BRM database.

  > **Note**
  >
  > This opcode overwrites data in existing **/plan** objects, so be sure you pass in the correct object to modify.

- When the PIN_FLD_DELETED_FLAG is set, this opcode *deletes* the specified **/plan** object by doing the following:

Copyright © 2025, 2026, Oracle and/or its affiliates.

  - Calling the PCM_OP_PRICE_POL_DELETE_DISCOUNT policy opcode to perform any custom validation. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.
  - Deleting the specified object in the BRM database.

PCM_OP_PRICE_COMMIT_PLAN performs all operations within a single transaction, so success or failure is indicated by the value returned for PIN_FLD_RESULTS:

- When successful, PCM_OP_PRICE_COMMIT_PLAN sets PIN_FLD_RESULTS to **1**.
- When any operation fails, PCM_OP_PRICE_COMMIT_PLAN_LIST sets PIN_FLD_RESULTS to **0**. It also returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

Managing /dependency Objects

Use the PCM_OP_PRICE_COMMIT_DEPENDENCY opcode to create, modify, or delete a **/dependency** object. This object specifies any dependencies between two discounts or between a discount and a plan.

This opcode accepts an flist that includes a PIN_FLD_DEPENDENCY array, each element of which represent a **/dependency** object.

Customizing How to Create and Delete Dependencies

To customize how to create and delete dependencies, use the following policy opcodes:

- To enable data modification during **/dependency** object creation, use the PCM_OP_PRICE_POL_PREP_DEPENDENCY policy opcode. Use this policy opcode to enhance **/dependency** objects with additional data not provided by either the GUI application or by the Price List FM. For example, you would modify this policy opcode if your business requires that you change a mutually exclusive relationship into a required one, without using Pricing Center.

  PCM_OP_PRICE_COMMIT_DEPENDENCY calls this policy opcode before creating a new **/dependency** object. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.

  This policy opcode is called before the Price List FM performs any validation checks.

  Success or failure is indicated by the value returned for PIN_FLD_RESULTS. When successful, PIN_FLD_RESULTS is set to **1**. When any operation fails, PIN_FLD_RESULTS is set to **0**.

- To enable validation during **/dependency** object creation, use the PCM_OP_PRICE_POL_VALID_DEPENDENCY policy opcode. This policy opcode can be used to change **/dependency** relationships without using Pricing Center.

  This policy opcode is called by PCM_OP_PRICE_SET_PRICE_LIST and PCM_OP_PRICE_COMMIT_DEPENDENCY before creating a **/dependency** object. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.

  Success or failure is indicated by the value returned for PIN_FLD_RESULTS. When successful, PIN_FLD_RESULTS is set to **1**. When any operation fails, PIN_FLD_RESULTS is set to **0**.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- To verify that deleting a **/dependency** object is permitted, use the PCM_OP_PRICE_POL_DELETE_DEPENDENCY policy opcode. Use this policy opcode to perform validations in addition to those performed by the Price List FM.

  For example, you can use this policy opcode to confirm that the valid time period for the **/dependency** object has expired before allowing it to be deleted.

  This policy opcode is called by PCM_OP_PRICE_COMMIT_DEPENDENCY before it deletes a **/dependency** object. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.

  Success or failure is indicated by the value returned for PIN_FLD_RESULTS. When successful, PIN_FLD_RESULTS is set to **1**. When any operation fails, PIN_FLD_RESULTS is set to **0**.

Flist /dependency Arrays

The arrays defining prerequisite and mutually exclusive relationships require the names of the two product objects. The name of the product that *requires* another product is listed in the PIN_FLD_DEPENDENT_NAME field; the name of the *required* product is listed in the PIN_FLD_DEPENDEE_NAME field. For example, if your business has an email product that requires a Basic Dialup product, **email** is the dependee and **Basic Dialup** is the dependent:

```
0 PIN_FLD_DEPENDENCIES   ARRAY  [12]
1  PIN_FLD_DEPENDEE_OBJ    POID [0] 0.0.0.1 /deal -1 0
1  PIN_FLD_DEPENDENT_OBJ   POID [0] 0.0.0.1 /deal -1 0
1  PIN_FLD_DEPENDEE_NAME    STR [0] "Basic Dialup"
1  PIN_FLD_DEPENDENT_NAME   STR [0] "email"
1  PIN_FLD_TYPE             ENUM [0] 1
```

PCM_OP_PRICE_COMMIT_DEPENDENCY performs all operations within a single transaction, so success or failure is indicated by the value returned for PIN_FLD_RESULTS:

- When successful, PCM_OP_PRICE_COMMIT_DEPENDENCY sets PIN_FLD_RESULTS to **1**.
- When any operation fails, PCM_OP_PRICE_COMMIT_DEPENDENCY sets PIN_FLD_RESULTS to **0**. It also returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

Managing /transition Objects

Use the PCM_OP_PRICE_COMMIT_TRANSITION opcode to create, modify, or delete a **/transition** object.

This opcode accepts an flist consisting of a PIN_FLD_TRANSITION array, each element of which represents a **/transition** object.

PCM_OP_PRICE_COMMIT_TRANSITION determines whether to create, modify, or delete the specified **/transition** object:

- When the object does not exist, this opcode *creates* the specified **/transition** object by doing the following:
  - Calling the PCM_OP_PRICE_POL_PREP_TRANSITION policy opcode to perform any custom preparation. By default, this policy opcode is an empty hook, but you can customize it to enhance **/transition** objects with additional data or to perform other custom actions.

Copyright © 2025, 2026, Oracle and/or its affiliates.

  - Calling the PCM_OP_PRICE_POL_VALID_DISCOUNT policy opcode to perform any custom validation. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.
  - Creating the specified object in the BRM database.
- When the object already exists, this opcode *modifies* the specified **/transition** object in the BRM database.

> **Note**
>
> This opcode overwrites data in existing **/transition** objects, so be sure you pass in the correct object to modify.

- When the PIN_FLD_DELETED_FLAG is set, this opcode *deletes* the specified **/transition** object by doing the following:
  - Calling the PCM_OP_PRICE_POL_DELETE_TRANSITION policy opcode to perform any custom validation. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.
  - Deleting the specified object in the BRM database.

PCM_OP_PRICE_COMMIT_TRANSITION performs all operations within a single transaction, so success or failure is indicated by the value returned for PIN_FLD_RESULTS:

- When successful, PCM_OP_PRICE_COMMIT_TRANSITION sets PIN_FLD_RESULTS to **1**.
- When any operation fails, PCM_OP_PRICE_COMMIT_TRANSITION sets PIN_FLD_RESULTS to **0**. It also returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

Customizing How to Create and Delete Transitions

To customize how to create and delete transitions, use the following policy opcodes:

- To enable data modification during **/transition** object creation, use the PCM_OP_PRICE_POL_PREP_TRANSITION policy opcode. Use this policy opcode to enhance **/transition** objects with additional data not provided by either the GUI application or by the Price List FM.

  For example, if your business waives purchase and cancellation fees for its San Francisco customers, you would implement that logic in this policy opcode.

  PCM_OP_PRICE_COMMIT_TRANSITION calls this policy opcode before creating a new **/transition** object. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.

  This policy opcode is called before the Price List FM performs any validation checks.

  Success or failure is indicated by the value returned for PIN_FLD_RESULTS. When successful, PIN_FLD_RESULTS is set to **1**. When any operation fails, PIN_FLD_RESULTS is set to **0**.
- To enable validation during **/transition** object creation, use the PCM_OP_PRICE_POL_VALID_TRANSITION policy opcode. This policy opcode can be used to change **/transition** relationships without using Pricing Center.

Copyright © 2025, 2026, Oracle and/or its affiliates.

For example, if your business requires that you prohibit underage customers from transitioning to a specific deal, you would add that limitation to this policy opcode.

This policy opcode is called by PCM_OP_PRICE_COMMIT_TRANSITION before creating a **/transition** object. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.

This policy opcode is called before the Price List FM creates the **/transition** object.

Success or failure is indicated by the value returned for PIN_FLD_RESULTS. When successful, PIN_FLD_RESULTS is set to **1**. When any operation fails, PIN_FLD_RESULTS is set to **0**.

- To verify that deleting a **/transition** object is permitted, use the PCM_OP_PRICE_POL_DELETE_TRANSITION policy opcode. Use this policy opcode to perform validations in addition to those performed by the Price List FM.

  For example, you can use this policy opcode to confirm that the valid time period for the **/transition** object has expired before allowing it to be deleted.

  This policy opcode is called by PCM_OP_PRICE_COMMIT_TRANSITION before it deletes a **/transition** object. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.

  Success or failure is indicated by the value returned for PIN_FLD_RESULTS. When successful, PIN_FLD_RESULTS is set to **1**. When any operation fails, PIN_FLD_RESULTS is set to **0**.

Managing /group/plan_list Objects

Use the PCM_OP_PRICE_COMMIT_PLAN_LIST opcode to commit **/group/plan_list** objects to the BRM database.

This opcode connects to the database, opens a transaction, and retrieves the **/plan_list** flist. PCM_OP_PRICE_COMMIT_PLAN_LIST searches the PIN_FLD_PLAN flist to match the name and type fields in the plan list flist. This plan is checked against existing plans in the database. If the plan does not exist, this opcode reports an error and stops the transaction. If the plan does exist, this opcode adds the flist to the PIN_FLD_RESULTS array. This process is repeated until all plan list entries are added to the PIN_FLD_RESULTS array and committed to the database. The committed changes include new, modified, or deleted objects.

PCM_OP_PRICE_COMMIT_PLAN_LIST performs all operations within a single transaction, so success or failure is indicated by the value returned for PIN_FLD_RESULTS:

- When successful, PCM_OP_PRICE_COMMIT_PLAN_LIST sets PIN_FLD_RESULTS to **1**.
- When any operation fails, PCM_OP_PRICE_COMMIT_PLAN_LIST sets PIN_FLD_RESULTS to **0**. It also returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

Managing /sponsorship Objects

Use the PCM_OP_PRICE_COMMIT_SPONSORSHIP opcode to create, change, or delete a **/sponsorship** object.

PCM_OP_PRICE_COMMIT_SPONSORSHIP determines whether to create, modify, or delete the specified **/sponsorship** object:

- When the object does not exist, this opcode *creates* the specified **/sponsorship** object by doing the following:

Copyright © 2025, 2026, Oracle and/or its affiliates.

  - Calling the PCM_OP_PRICE_POL_PREP_DISCOUNT policy opcode to perform any custom preparation. By default, this policy opcode is an empty hook, but you can customize it to enhance **/sponsorship** objects with additional data or to perform other custom actions.
  - Calling the PCM_OP_PRICE_POL_VALID_DISCOUNT policy opcode to perform any custom validation. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.
  - Calling the PCM_OP_SUBSCRIPTION_POL_VALIDATE_OFFERING policy opcode. By default, this policy opcode is an empty hook, but you can customize it to validate any product specification attributes defined for **/sponsorship** objects. See "Defining Product Specification Attributes for Pricing Components" for more information.
  - Creating the specified object in the BRM database.
  - Preparing a list of all **/sponsorship** objects that were created.
  - Generating the **/event/notification/price/sponsorships/modify** notification event.
- When the object already exists, this opcode *modifies* the specified object by doing the following:
  - Updating the specified object in the BRM database.
  - Preparing a list of all **/sponsorship** objects that were modified.
  - Generating the **/event/notification/price/sponsorships/modify** notification event.

> **Note**
>
> This opcode overwrites data in existing **/sponsorship** objects, so be sure you pass in the correct object to modify.

- When the PIN_FLD_DELETED_FLAG is set, this opcode *deletes* the specified **/sponsorship** object by doing the following:
  - Calling the PCM_OP_PRICE_POL_DELETE_DISCOUNT policy opcode to perform any custom validation. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.
  - Deleting the specified object in the BRM database.

PCM_OP_PRICE_COMMIT_SPONSORSHIP performs all operations within a single transaction, so success or failure is indicated by the value returned in the PIN_FLD_RESULTS output flist field: a value of **1** indicates that all operations succeeded and a value of **0** indicates that the operation failed. If the operation fails, the opcode also returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

Managing Tax Selectors and Tax Exemption Selectors

Tax selectors enable you to apply tax codes to your customers' products based on specified attributes in **/account**, **/service**, **/event**, and **/profile** objects. Each tax selector specifies:

- The name of the tax selector.
- The name of the tax code along with its associated criteria; that is, the list of event, service, profile, or account fields that an event must contain to qualify for the tax code.

Copyright © 2025, 2026, Oracle and/or its affiliates.

BRM stores information about tax selectors in **/tax_selector** objects, and tax exemptions in **/tax_exemption_selector** objects. For more information about these objects, see *BRM Storable Class Reference*.

To manage tax selectors, use the following opcodes:

- PCM_OP_PRICE_CREATE_SELECTOR. See "Creating Tax Selectors".
- PCM_OP_PRICE_GET_SELECTOR. See "Retrieving Tax Selectors".
- PCM_OP_RATE_EVALUATE_SELECTOR. See "Retrieving the Tax Code from a Selector".

Creating Tax Selectors

Use the PCM_OP_PRICE_CREATE_SELECTOR opcode to create a tax selector (a **/tax_selector** or **/tax_exemption_selector** object) in the BRM database. This opcode is called by PCM_OP_PRICE_SET_PRICE_LIST.

PCM_OP_PRICE_CREATE_SELECTOR performs the following operations:

1. Reads the selector rules and converts them into a buffer field.
2. If the PIN_FLD_POID input flist field is set to **0.0.0.***x* **/tax/selector -1 0**, does the following:
   a. Creates a **/tax_selector** object.
   b. Adds the **/tax_selector** POID to the **/rate_plan** object.
   c. Returns the POID of the new **/tax_selector** object in the opcode's output flist.
3. If the PIN_FLD_POID input flist field is set to **0.0.0.***x* **/tax_exemption_selector -1 0**, does the following:
   a. Creates a **/tax_exemption_selector** object.
   b. Adds the **/tax_exemption_selector** POID to the **/product** object.
   c. Returns the POID of the new **/tax_exemption_selector** object in the opcode's output flist.

Retrieving Tax Selectors

Use the PCM_OP_PRICE_GET_SELECTOR opcode to retrieve a tax selector (**/tax_selector** or **/tax_exemption_selector** object) from the BRM database. This opcode is called by PCM_OP_PRICE_GET_PRICE_LIST.

To retrieve a tax selector, pass one of the following in the opcode's input flist:

- PIN_FLD_POID set to a type-only POID and PIN_FLD_NAME set to the name of the tax selector. For example:

```
0 PIN_FLD_POID    POID [0] 0.0.0.1 /tax_selector -1 0
0 PIN_FLD_NAME     STR [0] "my_tax_selector"
```

- PIN_FLD_POID set to the tax selector object's POID. For example:

```
0 PIN_FLD_POID    POID [0] 0.0.0.1 /tax_exemption_selector 120916 0
```

The opcode returns the tax selector object POID along with details about the tax selector, such as the tax selector rules and tax codes.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Retrieving the Tax Code from a Selector

Use the PCM_OP_RATE_EVALUALTE_SELECTOR opcode to determine the correct tax code or tax exemption code to apply based upon the attributes provided. This opcode is called by PCM_OP_PRICE_GET_PRICE_LIST.

To determine the correct tax code or tax exemption code to apply, pass one of the following in the opcode's input flist:

- If pipeline is enabled, PIN_FLD_POID set to a type-only POID. For example:

```
0 PIN_FLD_POID    POID [0] 0.0.0.1 /tax_selector -1 0
```

- PIN_FLD_POID set to the tax selector object's POID. For example:

```
0 PIN_FLD_POID    POID [0] 0.0.0.1 /tax_exemption_selector 120916 0
```

PCM_OP_RATE_EVALUATE_SELECTOR performs the following operations:

1. Reads the event details and tax selector, and then converts them into a buffer field.
2. Compares the event details to the criteria for each tax code. If the event matches the event, service, profile, or account fields required for a tax code, the tax code is added to the output flist.
3. Calls PCM_OP_RATE_POL_POST_EVALUATE_SELECTOR to perform optional customization.
4. Returns the tax code or tax exemption code.

Managing Filter Sets

Filter sets enable you to apply system products and system discounts to a select group of customers (for example, those with the best credit history). You define which customers and events qualify for specified products and discounts by using Pricing Center or a custom client application. Each filter set specifies:

- The criteria an EDR must meet to qualify for the filter set: that is, list of required EDR fields and their values
- The list of applicable system products and system discounts, along with their validity dates and priorities

BRM stores information about each filter set in **/filter_set/product** objects.

To manage filter sets, use the following opcodes:

- PCM_OP_FILTER_SET_CREATE. See "Creating Filter Sets".
- PCM_OP_FILTER_SET_UPDATE. See "Updating Filter Sets".
- PCM_OP_FILTER_SET_DELETE. See "Deleting Filter Sets".

Creating Filter Sets

Use the PCM_OP_FILTER_SET_CREATE opcode to create **/filter_set/product** objects in the BRM database. This opcode is called directly by Pricing Center.

PCM_OP_FILTER_SET_CREATE performs the following operations:

Copyright © 2025, 2026, Oracle and/or its affiliates.

1. Confirms that the object to be created is a **/filter_set/product** object, or one subclassed from it. The type of object is specified in the PIN_FLD_POID field of the input flist.
2. Creates the **/filter_set/product** object.
3. Generates an **/event/filter_set/create** object to record details of the event.
4. Returns the POID of the new **/filter_set/product** object.

Updating Filter Sets

Use the PCM_OP_FILTER_SET_UPDATE opcode to modify existing **/filter_set/product** objects. This opcode is called directly by Pricing Center.

PCM_OP_FILTER_SET_UPDATE performs the following operations:

1. Determines whether to add, modify, or delete data in the object by checking the PIN_FLD_FLAG field:
   - When the flag is set to **0**, the opcode *modifies* filter set data.
   - When the flag is set to **1**, the opcode *adds* filter set data.
   - When the flag is set to **2**, the opcode *deletes* filter set data.
2. Modifies data in the specified **/filter_set/product** object.
3. Creates an **/event/filter_set/update** object to record details of the event.
4. Returns the POID of the **/filter_set/product** object.

Deleting Filter Sets

Use the PCM_OP_FILTER_SET_DELETE opcode to delete **/filter_set/product** objects from the BRM database. This opcode is called directly by Pricing Center.

PCM_OP_FILTER_SET_DELETE performs the following operations:

1. Deletes the specified **/filter_set/product** object.
2. Generates an **/event/filter_set/delete** object to record details of the event.
3. Returns the POID of the **/filter_set/product** object.

Copyright © 2025, 2026, Oracle and/or its affiliates.

14

Testing Your Price List

This chapter describes the procedure for testing an Oracle Communications Billing and Revenue Management (BRM) price list. Testing is an important step to take before using the price list for customers.

Topics in this document:

- About Using a Test Database
- Creating Test Accounts
- Generating Test Account Activity
- Advancing the Date
- Generating a Test Invoice
- Example Price List Test.

For information about creating a price list, see "About Creating a Price List".

About Using a Test Database

Before testing a price list, set up a test database that is separate from your production database. Because you simulate account activity and time changes when you test your price list, you would create erroneous events in a production BRM database that contains real customer accounts.

> **Note**
>
> You can remove all price list elements in a database by using the **loadpricelist utility -p** parameter. See "loadpricelist".

Creating Test Accounts

Use Customer Center to create test accounts in your test database. Note the login name for each account you create, because you will need it later.

> **Note**
>
> To create accounts with your new plans, you must add them to a plan list.

When selecting a payment method, you can select either Invoice or Credit Card. If you select Credit Card, also do the following:

- Ensure that the necessary Data Managers and applications are properly configured and running:

Copyright © 2025, 2026, Oracle and/or its affiliates.

  - For Paymentech, **dm_fusa**, **answer_b**, and **answer_s** should be running.

- You must use one of the following pairs of credit card numbers and expiration dates listed in Table 14-1 for your test accounts:

Table 14-1 Example Credit Card Expiration Data

Credit Card Number	Expiration Date
4444 1111 2222 3333	0999
4101 0000 0000 0000	any expiration date

> **Note**
>
> A simple cycle rate, such as a monthly fee for a service, should appear in an account's balance as soon as you create the account.

Generating Test Account Activity

Use the **sample_act** utility to generate activity for a test account. You use this utility to simulate a customer using a product or service without the product or service actually being available. You can then see if the activity changes the test account's balance as you expect.

Perform these tasks to generate activity with **sample_act**:

- Running sample_act
- Checking Results of Running sample_act

Running sample_act

To run the **sample_act** utility:

1. Go to *BRM_home*/**source/apps/sample**.
2. Edit the **pin.conf** file for the **cm_ptr** to point to your CM process. Also, edit the login type, login name, and password entries.
3. Run the **sample_act** utility for each account you want to include in your test.

   The syntax for **sample_act** depends on the type of event you are simulating:

   - **Session event**: A usage event, such as hours of Internet access. If the rate depends on the passage of time, use a session event.

     For a session event, use this syntax:

     ```
     sample_act -v -e session -l login -d duration -s servicename
     ```

   - **Activity event**: An item or cycle fee that does not depend on time, such as a monthly fee.

     For an activity event, use this syntax:

     ```
     sample_act -v -e activity -l login -q quantity -s servicename
     ```

   Following is a description of the most common **sample_act** options:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- **-v**: Use to run **sample_act** in verbose mode. In this mode, you see status messages at the command prompt, including messages that the event generated successfully or that an error occurred. There is no argument.
- [**-n RUM** *Name* **-q** *quantity* [**-u** *Unit*]]
- [**-b** *event time*] (as MM:DD:YY:HH:MM:SS or UNIX time)
- [**-A** *ANI*] [**-D** *DNIS*] [**-Y** *call_type*]: ANI is the source telephone number and DNIS is the destination telephone number
- [**-X** *int_indicator*] [**-O** *origin_gw*]
- [**-N** *new Timezone ID String*]
- [**-I** *termserv_id*] [**-P** *termserv_port*] [**-T** *old timezone offset*]
- **-e** *activity*|*session* [**-c** *custom_subtype*]: Enter activity or session.
- **-l** *login*: Enter the login name for the test account and service for which you want to generate activity.
- **-q** *quantity*: For an activity event, enter the quantity purchased.
- **-d** *duration*: For a session event, enter the duration of the simulated event in seconds. The default is 3600 seconds, or one hour.
- **-s** *service_type* **-l** *service_login*: Enter the name of the service for which you are simulating the event, such as **/service/ip**.

The following example simulates using three hours of Internet service. The login is **pin_user** and the service is **/service/ip**.

```
sample_act -v -e session -l pin_user -d 10800 -s /service/ip
```

To test a rate that changes at different times, such as an Internet service rate with peak and off-peak rates, you can advance the system date in conjunction with running **sample_act**. See "Advancing the Date ".

Before advancing the time, you must edit the **sample_act** configuration file so the utility responds to the changed time. See "Setting Up the pin_virtual_time Utility".

Checking Results of Running sample_act

To get the results of your test after running **sample_act**:

1. Find the test accounts in Customer Center.
2. Click the **Balance** tab.
3. For detailed information about balance changes, use Event Browser.

Advancing the Date

To properly test a price list, you must simulate the passage of time. For example, you should test:

- If accounts are billed correctly. Advance the date to the next billing cycle, then run billing and generate an invoice.
- If a time-based discount takes effect correctly. If you have a deal that includes 90 days of free email service, for example, advance the date more than 90 days to see if the account starts being charged for the service.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- If folds are working correctly. If a product includes 10 free hours of Internet service per month, for example, advance the date by a month or more to ensure that unused free hours do not carry over.

The **pin_virtual_time** utility enables you to simulate changes in the BRM system time.

> **Note**
>
> - If you use **pin_virtual_time** with a BRM system that includes Pipeline Manager, you must set the **VirtualTime** parameter for the DAT_BalanceBatch module to **True** to ensure that balances are calculated correctly.
> - The **pin_virtual_time** utility works only on a single computer. Typically, you set up a test BRM system on just one computer. If you run **pin_virtual_time** on a system that is distributed across multiple computers, you must carefully coordinate time changes on all the computers.

Perform the following tasks to advance the date:

- Setting Up the pin_virtual_time Utility
- Running pin_virtual_time
- Stopping and Starting BRM

Setting Up the pin_virtual_time Utility

To set up the **pin_virtual_time** utility:

1. Go to the *BRM_home***/sys/test** directory.
2. Run this command to create a file that **pin_virtual_time** requires, called **pin_virtual_time_file**:

   ```
   pin_virtual_time -m 0 -f BRM_home/bin/pin_virtual_time_file
   ```

   > **Note**
   >
   > *BRM_home***/lib/pin_virtual_time_file** is the standard path and file name, but you can change it.

3. Add the following entry to the configuration file in the *BRM_home***/sys/test** directory:

   ```
   - - pin_virtual_time  pin_virtual_time_file
   ```

   Replace *pin_virtual_time_file* with the path and name of the mapped file created in the previous step.

   Adding the entry to the configuration file in **sys/test** enables you to run **pin_virtual_time** from that directory. The directory from which you run **pin_virtual_time** must contain a configuration file with the **pin_virtual_time** entry.

4. Add the entry in step 3 to the configuration file for each program you want to respond to an altered time.

   To test your price list, edit the configuration files for at least the applications listed in Table 14-2.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 14-2 Configuration Files for Testing Applications

Application	Configuration File Location
CM	**sys/cm/**
Oracle DM	**sys/dm_oracle/**
Billing and invoice utilities	**apps/pin_billd/**
sample_act utility	**source/apps/sample**

To ensure that all applications respond to **pin_virtual_time** changes, add the entry to all the configuration files in your test BRM system.

Entry example:

```
- - pin_virtual_time BRM_home/bin/pin_virtual_time_file
```

Running pin_virtual_time

After you set up the **pin_virtual_time** utility, run **pin_virtual_time** to advance the BRM date and time.

Syntax for **pin_virtual_time**:

```
pin_virtual_time -m 2 MMDDHHMM[CC]YY.[SS]
```

For the string MMDDHHMM[CC]YY.[SS], enter the date and time you want BRM to use in this format: month, date, hour, minute, year, and seconds. You must enter at least two digits for the year, but you can enter four. Seconds are optional.

For example, to set the date and time to 9/3/99 and 11:30, respectively:

```
pin_virtual_time -m 2 090311301999.00
```

The command displays this message at the command prompt:

```
filename BRM_home/lib/pin_virtual_time_file, mode 2, time: Fri Sept 03 11:30:00 1999
```

The time then advances normally from the new reset time.

Stopping and Starting BRM

After you run **pin_virtual_time**, you must stop and restart BRM to read in the new time:

1. Log on to the computer that is the server for your test BRM system. Usually, your test system should be on a single machine.

   > **Note**
   >
   > Log in as something other than **root**.

2. Exit all BRM client tools, such as Customer Center and Pricing Center.
3. Stop and restart the CM and Oracle Data Manager.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Generating a Test Invoice

Generate invoices for your test accounts to see if the accounts are billed correctly. Before following this procedure, you should have generated account activity and advanced the date to the beginning of the next billing cycle. Go to the **Payment** tab in Customer Center to find the Next Billing Start date for an account.

To run billing and generate a test invoice:

1. Go to the *BRM_home***/apps/pin_billd** directory.
2. At the command prompt, enter the following:

   ```
   pin_bill_day
   ```

   This command is a script that runs several billing utilities.

   BRM creates an invoice for each active account.
3. Open a test account in Customer Center and go to the **Payment** tab.
4. Ensure that the Current Billing Start and End dates are what you expect.
5. Search for the invoice for the current account, and see that it contains the results of your test. See "Displaying Invoices" in *BRM Designing and Generating Invoices*.

Example Price List Test

This example takes you through the steps for testing a plan that includes the following:

- A monthly Internet access fee of $9.95.
- An hourly fee of $2.00 for prime-time Internet service usage. Prime time is 8:00 a.m. to 6:00 p.m. weekdays.
- An hourly fee of $1.00 for off-peak Internet service usage. This includes nights and weekends.

You can test this plan as follows:

1. In Customer Center, create a test account by using the plan you are testing.
2. Go to the **Balance** tab. The current balance for the US Dollar resource should be $9.95, because the monthly access fee impacts the balance right away.
3. Close the test account.
4. Set up the **pin_virtual_time** utility, by using the procedure in "Setting Up the pin_virtual_time Utility". Ensure that you at least add the **pin_virtual_time** entry to the configuration files for the Connection Manager, Oracle Data Manager, and **sample_act** utility.
5. Go to the *BRM_home***/sys/test** directory.
6. Change the BRM system time so it is close to the time when one rate ends and the other begins. For example, set the time to 6:00 a.m. the following day.

   If you created the account on May 1, 1999, you would run the following command to change BRM's system time to 6:00 a.m. the next morning:

   ```
   pin_virtual_time -m 2 0502060099 -l pin_user
   ```

   In the above command, *pin_user* is the login assigned to the test account.

Copyright © 2025, 2026, Oracle and/or its affiliates.

7. Stop and restart BRM, by using the procedure in "Stopping and Starting BRM".
8. Go to *BRM_home*/**source/apps/sample**.
9. Generate four hours of Internet access activity for the account by running this command:

   ```
   sample_act -v -e session -r ip/dialup/async/hourly -s /service/ip -d 14400 -l
   pin_user
   ```

   In the above command, **ip/dialup/async/hourly** is the rate category, **/service/ip** is the service, 14400 is four hours in seconds, and **pin_user** is the login name for the account.
10. Reopen the account in Customer Center.
11. Go to the **Balance** tab.

    The current balance for the US Dollar resource should now be $15.95. The **sample_act** command should have generated two hours of Internet access at the off-peak fee of $1 and two hours at the prime-time fee of $2. If you set up the rates correctly, this adds $6.00 to the previous balance of $9.95.

Copyright © 2025, 2026, Oracle and/or its affiliates.

15

Using the XML Pricing Interface to Create a Price List

This chapter describes how to create and edit an Oracle Communications Billing and Revenue Management (BRM) price list by using the XML Pricing Interface.

Topics in this document:

- About Creating and Modifying Price List Files
- About the XML Pricing Interface
- Starting the XML Pricing Interface
- Creating a Price List with the XML Pricing Interface
- Modifying Existing Price Lists with the XML Pricing Interface
- Moving a Price List between Databases with the XML Pricing Interface
- Downloading Price Lists from the Database with the XML Pricing Interface
- Loading a Price List into the Database with the XML Pricing Interface
- Purging Price Lists from the BRM Database

About Creating and Modifying Price List Files

You create and modify price lists by using the following:

- **Pricing Center**: Pricing Center is a GUI application that provides on-screen wizards to help you create your price list. You create and modify the price list and retrieve and save it to the database all directly in Pricing Center. For more information, see Pricing Center Help.
- **XML Pricing Interface**: The XML Pricing Interface consists of the "loadpricelist" utility. You create price lists directly in an XML editor or text editor and then use the utility to load, retrieve, and delete the price list from the BRM database. For information on creating price lists with the XML Pricing Interface, see "About the XML Pricing Interface".

About the XML Pricing Interface

The XML Pricing Interface is a command-line utility called **loadpricelist** that you can use to download, edit, and load price lists.

> **Note**
>
> The XML price list must follow the format detailed in the XML SchemaDefinition (XSD) file **price_list.xsd**. By default, the **price_list.xsd** file is installed in *BRM_home*/**PricingCenter** and/or *BRM_home*/**setup/scripts**.

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Downloading and Loading XML Price List Files

You can use the **loadpricelist** utility to download and load an entire price list or a partial price list. When you download a price list from the database, the utility writes the price list information you specify into an XML file. If you specify to download to an existing price list file, the utility overwrites the entire XML file.

When you load a price list from an XML file into the database, the utility modifies only those objects that have changed:

- If an object has changed, the utility updates the object's attributes in the database.
- If an object does not exist in the database, the utility creates the database object.
- If an object has not changed and is already in the database, the utility does not modify the database object.

Prerequisites for the XML Pricing Interface

Before you use the XML Pricing Interface, you must configure the following:

- Include the **loadpricelist.jar**, **xerces.jar**, **pcm.jar**, and **pcmext.jar** files in your CLASSPATH. The **xerces.jar**, **pcm.jar**, and **pcmext.jar** files are located in the **C:\Program Files\Common Files\Portal Software** directory and subdirectories by default.

  > **Note**
  >
  > Instead of including the JAR files in your CLASSPATH, you can include them in the **loadpricelist** file.

- Include the location of the **Infranet.properties** file in your CLASSPATH.

  The default **Infranet.properties** file is in the **\common** directory. To put the **Infranet.properties** file in the local directory with the **loadpricelist** utility, add **./** (period-slash) to the beginning of the **Infranet.properties** CLASSPATH.

- Ensures that the **Infranet.properties** file specifies the correct login information for the BRM database you want to connect to.

  > **Note**
  >
  > You might need to change this information if, for example, you want to download a price list from one BRM database, modify it, and load it to a different BRM database.

- Install Pricing Center or copy the **price_list.xsd** file to your system. You can find the **price_list.xsd** file on any BRM server in either the *BRM_home***/PricingCenter** directory or the *BRM_home***/setup/scripts** directory.

Starting the XML Pricing Interface

To start the XML Pricing Interface, at the command prompt, enter **loadpricelist** followed by an interactive or non-interactive command.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Getting Help for the XML Pricing Interface

To get help for the XML Pricing Interface:

1. Start the XML Pricing Interface, if necessary.
2. Enter **help** at the command prompt and press **Enter**.

Working in Interactive and Non-Interactive Mode

You can work in interactive and non-interactive mode in the XML Pricing Interface.

- In interactive mode, you issue a command for each action. For a list of commands that can be used in this mode, see "Syntax: Interactive Mode".
- In non-interactive mode, you can use special commands that batch several related actions together so the system performs them automatically. For a list of parameters that can be used in this mode, see "Syntax: Non-Interactive Mode".

You receive a confirmation message when the XML Pricing Interface carries out a command successfully. If you do not receive a confirmation message, look in the utility log file (**javapcm.log**) to find any errors. The log file is either in the directory where the utility was started or in a directory specified in the **Infranet.properties** file.

> **Note**
>
> If an error occurs during a command, none of the command is carried out even if the error was only caused by part of the operation.

Creating Price Lists with the XML Pricing Interface

You create price lists in an XML editor or text editor and then load it into the BRM database by using the "loadpricelist" utility.

To create a price list:

1. Plan your price list. You should know in advance how you will set up your rates, products, deals, and plans. Ensure that the following pricing objects and data have been created in the BRM database:
   - services
   - events
   - resources
   - currency exchange rates
   - G/L IDs
   - tax codes and tax suppliers
   - ratable usage metrics (RUMs)
   - product-level provisioning categories
   - zones

   For information on planning your price list, see "About Creating a Price List".

Copyright © 2025, 2026, Oracle and/or its affiliates.

2. Create an XML file, such as **price_list.xml**, in a text editor or XML editor.
3. Enter your price list information. Ensure that your price list follows the structure defined in the **price_list.xsd** file.

> **Note**
>
> You can view sample XML price list files in the *BRM_home***PricingCenter\Sample_Price_Plans** directory.

   See "XML Examples of Creating Pricing Components" for some pricing configuration examples using XML.
4. Save and close the file.
5. Ensure that any billable events in your price list are also included in the **pin_event_map** file. Otherwise, they will not be rated. See "Mapping Events and Services".
6. Load the price list into the BRM database. See "Loading a Price List into the Database with the XML Pricing Interface".

BRM can begin using the price list after it is committed to the database.

Modifying Existing Price Lists with the XML Pricing Interface

To modify an existing price list file:

1. Retrieve the existing price list file. The method you use depends on the file's location:
   - **The XML Pricing Interface**: Your price list file is saved in XML format. You can open it directly in an XML editor or text editor.
   - **The BRM database**: You must download the price list to your system and write the information to an XML file. See "Downloading Price Lists from the Database with the XML Pricing Interface".
2. Open the XML price list file in a text editor or XML editor.
3. Edit your price list file. Ensure that it follows the structure defined in the **price_list.xsd** file. By default, **price_list.xsd** is located in *BRM_home*/**PricingCenter**.

> **Note**
>
> You can view sample XML price list files in the *BRM_home***PricingCenter\Sample_Price_Plans** directory.

4. Save and close the file.
5. Ensure that any billable events in your price list are also included in the **pin_event_map** file. Otherwise, they will not be rated. See "Mapping Events and Services".
6. Load the price list into the BRM database. See "Loading a Price List into the Database with the XML Pricing Interface".

BRM can begin using the price list after it is committed to the database.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Moving a Price List between Databases with the XML Pricing Interface

You can use the **loadpricelist** utility to move a price list from one database to another (for example, from a test database to a production database). You move the price list by downloading it from a *source* BRM database and then loading it into a *destination* BRM database.

To move a price list from one database to another:

1. Download the price list from the source database. See "Downloading Price Lists from the Database with the XML Pricing Interface".
2. If necessary, edit the XML file in a text editor or XML editor.
3. Load the price list into the destination BRM database. See "Loading a Price List into the Database with the XML Pricing Interface".

Downloading Price Lists from the Database with the XML Pricing Interface

You can download either the entire price list or just a subset of the price list from the BRM database into an XML file that can be edited.

> **Note**
>
> Price lists are stored in the database in flist format. The utility converts the file into XML format as part of the download process.

Downloading a Full Price List

To download a full price list:

1. Ensure that the **Infranet.properties** file in your CLASSPATH has the correct login information for the BRM database you want to download price list information from. See "Setting global options" in *BRM Developer's Guide*.
2. Retrieve the price list from the BRM database and write it to an XML file:

   Interactive mode

   ```
   loadpricelist
   retrieve
   write [PriceListFile]
   ```

   where *PriceListFile* specifies the name and location of the file to which to extract the data.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> You can retrieve only one price list at a time. If you retrieve another price list when one is already in internal memory, the existing internal flist is overwritten.

Non-interactive mode

```
loadpricelist [-f] -r PriceListFile
```

where *PriceListFile* specifies the name and location of the file to which to extract the data.

Your price list is written to the specified XML file. You can now edit it directly in an XML editor.

Downloading a Subset of the Price List

You can write specific pricing objects, such as **/product**, **/discount**, or **/sponsorship** objects, rather than the entire price list to the XML price list file.

To download a subset of the price list:

1. Ensure that the **Infranet.properties** file in your CLASSPATH has the correct login information for the BRM database you want to connect to.
2. Specify the pricing objects to retrieve according to their service type, modification time, or object type, and write them to a specified XML file.

   Interactive mode

   ```
   loadpricelist
   retrieve [-s ServiceType][-t ModifiedTime][product][discount][sponsorship]
   write PriceListFile
   ```

   where:

   - **-s** *ServiceType* specifies to retrieve the specified service type. You can list multiple service types by using a comma (**,**) as a delimiter.

     > **Note**
     >
     > Do not use this parameter with the **sponsorship** parameter.

   - **-t** *ModifiedTime* retrieves objects that were modified after the specified timestamp.

     The time is specified in the ISO-8601 format: *YYYY***-***MM***-***DD***T***hh***:***mm***:***ss*TZD, with the server's local time zone as the default. For example:

     ```
     -t 1997-07-16T19:20:30+01:00
     ```

   - **product** specifies to retrieve only the **/product** objects from the database.
   - **discount** specifies to retrieve only the **/discount** objects from the database.
   - **sponsorship** specifies to retrieve only the **/sponsorship** objects from the database.
   - *PriceListFile* specifies the name and location of the file to which to extract the data.

   Non-interactive mode

   ```
   loadpricelist [-f] -r PriceListFile [-P][-D][-S][-s ServiceType][-t ModifiedTime]
   ```


Copyright © 2025, 2026, Oracle and/or its affiliates.

where:

- **f** specifies to run the command without prompting for a confirmation.
- *PriceListFile* specifies the name and location of the file to which to extract the data.
- **-P** specifies to retrieve only the **/product** objects from the database.
- **-D** specifies to retrieve only the **/discount** objects from the database.
- **-S** specifies to retrieve only the **/sponsorship** objects from the database.
- **-s** *ServiceType* retrieves objects based on the specified service type. You can list multiple service types by using a comma (**,**) as a delimiter.

  > **Note**
  >
  > Do not use this parameter with the **-S** parameter.

- **-t** *ModifiedTime* retrieves objects that were modified after the specified timestamp.

  The time is specified in the ISO-8601 format: *YYYY***-***MM***-***DD***T***hh***:***mm***:***ss*TZD, with the server's local time zone as the default. For example:

  ```
  -t 1997-07-16T19:20:30+01:00
  ```

Downloading Price Lists in Batches

If you have a large price list, downloading it by using **loadpricelist** can require more than the available system memory. In this situation, you should configure **loadpricelist** to retrieve and write pricelists in batches. When configured this way, **loadpricelist** performs a step search and writes the results into multiple XML files based on the batch size you specify.

You use the **infranet.batchsize** entry in the **loadpricelist Infranet.properties** file to configure the batch size. The batch size is expressed as the number of pricing objects per batch. For example, the following entry configures **loadpricelist** to retrieve and write pricing objects in batches of 500.

```
infranet.batchsize=500
```

When a batch size is configured, **loadpricelist** creates files based on the file name you specify in the command and on the type of pricing object, incrementing the name appropriately for each file created. For example, if the batch size is set to 500 and there are 1900 pricing objects in the database, then **loadpricelist** creates four XML files, the first three with 500 pricing objects and one with the remaining 400 objects.

When retrieving pricing objects in batches, you cannot choose which type of pricing object to retrieve; for example, only products, or only discounts. The -P, -D, -S, -s, and -t parameters are ignored.

In this scenario, this command:

```
loadpricelist -rf MyPriceList
```

results in the following four files being created (along with others for the other pricing objects in the database):

MyPriceList_product_1.xml

MyPriceList_product_2.xml

Copyright © 2025, 2026, Oracle and/or its affiliates.

MyPriceList_product_3.xml

MyPriceList_product_4.xml

File naming follows the same pattern for all object types. For example, file names for **/discount** objects would start with **MyPriceList_discount_1.xml** and increment from there.

The **batchsize** entry is not included in the **loadpricelist** file by default. To set the batch size:

1. Open the **Infranet.properties** file for **loadpricelist** in a text editor.

   The default **Infranet.properties** file is in the **\common** directory, but may be located in the local directory with the **loadpricelist** utility**.**

2. Add the **batchsize** entry to the file using the following syntax:

   ```
   infranet.batchsize=batch_size
   ```

   where *batch_size* represents the number of pricing objects to include in each batch.

3. Save and close the file.

Loading a Price List into the Database with the XML Pricing Interface

After your price list is complete, you load it into the BRM database. The XML Pricing Interface automatically converts the XML price list into flist format when it is loaded into the database.

> **Note**
>
> The price list file must be in the same directory as the **price_list.xsd** file. If they are not in the same directory, edit the **noNamespaceSchemaLocation** entry in the price list file to include the full path to **price_list.xsd**. By default, **price_list.xsd** is in the *BRM_home***PricingCenter** directory. For example, if the XSD file is located in **C:\pricelists**, change the entry to:
> **xsi:noNamespaceSchemaLocation="C:\pricelists\price_list.xsd"**

To load a price list:

1. Ensure that the XML price list file is complete and follows the guidelines specified in the XSD file.
2. Ensure that the **Infranet.properties** file in your CLASSPATH has the correct login information for the BRM database you want to connect to.
3. Commit the price list XML file to the BRM database.

   Interactive mode

   ```
   loadpricelist
   read PriceListFile
   commit
   ```

   where *PriceListFile* specifies the name and location of the file to load into the database.

   Non-interactive mode

   ```
   loadpricelist [-f] -c PriceListFile
   ```


Copyright © 2025, 2026, Oracle and/or its affiliates.

where:

- **f** specifies to run the command without prompting for a confirmation.
- *PriceListFile* specifies the name and location of the file to load into the database.

Purging Price Lists from the BRM Database

You can purge the following from the BRM database:

- All price list objects. See "Purging the Entire Price List from the BRM Database".
- A subset of the price list, based on the object type. See "Deleting a Subset of the Price List from the BRM Database".
- A specific price list object. See "Deleting Pricing Objects from the BRM Database".
- All dependency or transition objects. See "Purging Dependency or Transition Objects from the BRM Database".

Purging the Entire Price List from the BRM Database

To purge the entire price list from the database:

1. Ensure that the **Infranet.properties** file has the correct login information for the BRM database.
2. Purge the price list from the BRM database:

   Interactive mode

   ```
   loadpricelist
   purge
   ```

   Non-interactive mode

   ```
   loadpricelist [-f] -p
   ```

   where **f** specifies to run the command without prompting for a confirmation.

Deleting a Subset of the Price List from the BRM Database

> **Note**
>
> To delete a subset of price list data from the BRM database, you must use interactive mode.

To delete a subset of the price list:

1. Ensure that the **Infranet.properties** file has the correct login information for the BRM database.
2. Start **loadpricelist** in interactive mode and read *PriceListFile* into internal memory.

   ```
   loadpricelist
   read PriceListFile
   ```

3. Specify the pricing data to delete:

Copyright © 2025, 2026, Oracle and/or its affiliates.

```
delete [planlist[new|addon] | plan | bestpricing | deal | product | discount |
sponsorship]
```

where:

- **planlist** [**new|addon**] specifies to delete the plan list information from the BRM database. You can optionally specify to delete only the **new** plan lists or the **addon** plan lists.
- **plan** specifies to delete only the **/bestpricing** objects from the BRM database.
- **deal** specifies to delete only the **/deal** objects from the BRM database.
- **product** specifies to delete only the **/product** objects from the BRM database.
- **discount** specifies to delete only the **/discount** objects from the BRM database.
- **sponsorship** specifies to delete only the **/sponsorship** objects from the BRM database.

Deleting Pricing Objects from the BRM Database

> **Note**
>
> To delete individual pricing objects from the BRM database, you must use interactive mode.

To delete pricing objects:

1. Ensure that the **Infranet.properties** file has the correct login information for the BRM database.
2. Start **loadpricelist** in interactive mode and read *PriceListFile* into internal memory.

   ```
   loadpricelist
   read PriceListFile
   ```
3. Specify the pricing object to delete.

   ```
   delete PriceObject
   ```

   where *PriceObject* specifies the pricing object to delete.

Purging Dependency or Transition Objects from the BRM Database

> **Note**
>
> To delete only dependency or transition objects from the BRM database, you must use interactive mode.

To delete dependency or transition objects:

1. Ensure that the **Infranet.properties** file has the correct login information for the BRM database.
2. Specify the objects to delete.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- To delete only the dependency objects, enter:

```
loadpricelist
delete dependency
```

- To delete only the transition objects, enter:

```
loadpricelist
delete transition
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

16

XML Examples of Creating Pricing Components

This chapter describes how to configure pricing components in Oracle Communications Billing and Revenue Manager (BRM) by using an XML file and the **loadpricelist** utility.

Topics in this document:

- Prorating Fees for Billing DOM Changes
- Setting Proration for Products in a Deal
- Configuring Subscription Terms
- Setting Full Day Proration
- Adding Price Tags to Products and Discounts
- Allowing Customers to Exceed Their Credit Limit
- Transitioning Plans and Deals
- Configuring Add-On Products in Deals
- Purchasing the Same Product or Discount Multiple Times
- Aligning Recurring Charges and Product Validity to a Specific Day of the Month
- Splitting Noncurrency Balances into Multiple Validity Periods
- Setting Discount Validity Rounding
- Setting Product Cycle Alignment for Reactivated Deals
- Activating Products in Plans and Deals on First Usage
- Using Date Ranges for Versioning
- Defining Product Specification Attributes for Pricing Components
- Stop Rating Inactive, Canceled, or SuspendedActive Accounts
- Setting Loan Thresholds for Plans
- Enabling Dynamic Credit Floors in Plans
- Creating Tax Selectors for Products
- Creating Tax Exemption Selectors for Products

The following topics show examples of creating pricing components using XML elements in your price list file.

Prorating Fees for Billing DOM Changes

When configuring rates in XML, you can specify how to apply cycle fees when customers change their billing day of the month (DOM) in the middle of their billing cycle. For example, on June 15, a customer changes his billing DOM from the 10th to the 30th, generating a partial month. You can specify whether the June 10 through June 30 bill contains the entire monthly cycle fee, a prorated monthly cycle fee, or no monthly cycle fee.

Copyright © 2025, 2026, Oracle and/or its affiliates.

To do so, set **proration_flag** in the **<rate>** element to one of the following:

- **full**: Applies the total cycle fees.
- **prorate**: Prorates the cycle fees. This is the default.
- **none**: Applies no cycle fees.

The following sample XML entries specify to prorate cycle fees when customers change their billing DOM during a billing cycle:

```
<rate type="default" proration_flag="prorate">
   <description>$9.95 a month</description>
   <step_resource_id>840</step_resource_id>
   <quantity_tier>
      <balance_impact scaled_unit="none" flag="sponsorable discountable
proratable">
         <resource_id>840</resource_id>
         <glid>102</glid>
         <fixed_amount>10.0</fixed_amount>
         <scaled_amount>0</scaled_amount>
      </balance_impact>
   </quantity_tier>
</rate>
```

Setting Proration for Products in a Deal

When configuring deals in XML, you can specify how each product in the deal calculates prorated cycle fees. For example, a deal could contain one product or discount that prorates based on 30-day months and another that is based on the actual number of days in the month.

To set proration at the product level in your deals, set **flags** in the **<deal_product>** element to one of the following:

- **prorate_30_day**: Prorated cycle fees are calculated based on a 30-day month, regardless of the number of days in the billing cycle. For example, if a deal was owned for six days in a cycle, the prorated fee would be the cycle fee multiplied by 0.20 (6 ÷ 30).
- **prorate_days_in_month**: Prorated cycle fees are calculated based on the number of days in a particular month, such as 28 days in February, 31 days in March, and 30 days in April. For example, if a deal was owned for six days in March, the prorated fee would be the cycle fee multiplied by 0.19 (6 ÷ 31).
- **prorate_system**: Prorated cycle fees are calculated according to the system-wide setting in your CM **pin.conf** file. See "Enabling 30-Day-Based Proration" in *BRM Configuring and Running Billing*.

> **Note**
>
> It is applicable only on products and does not affect discounts as discounts are always relative to products and therefore not subject to proration settings.

The following sample XML entries create a deal named **Sample Deal** with products **Product01** and **Product02**. **Product01** specifies to prorate all of its cycle fees using a 30-day month,

Copyright © 2025, 2026, Oracle and/or its affiliates.

Product02 specifies to prorate all of its cycle fees based on the actual number of days in the month:

```
<deal customization_flag="optional" ondemand_billing="no">
   <deal_name>Sample Deal</deal_name>
   <description>My Sample Deal</description>
   <permitted>/service/ip</permitted>
   <deal_product flags="prorate_30_day">
      <product_name>Product01</product_name>
      <product_code>My Sample Product</product_code>
      <quantity>1.0</quantity>
      <purchase_discount>0.0</purchase_discount>
      <cycle_discount>0.0</cycle_discount>
      <usage_discount>0.0</usage_discount>
      <purchase_mode>0</purchase_mode>
   <deal_product flags="prorate_days_in_month">
      <product_name>Product01</product_name>
      <product_code>Another Sample Product</product_code>
      <quantity>1.0</quantity>
      <purchase_discount>0.0</purchase_discount>
      <cycle_discount>0.0</cycle_discount>
      <usage_discount>0.0</usage_discount>
      <purchase_mode>0</purchase_mode>
   </deal_product>
```

You can modify the **flags** value by using the PCM_OP_PRICE_SET_PRICE_LIST opcode. However, after customers subscribe to your deals, further updates to **flags** do not affect any purchased products.

You can override the **flags** value for a specific customer using the PCM_OP_SUBSCRIPTION_PURCHASE_DEAL or PCM_OP_CUST_COMMIT_CUSTOMER opcode. See "Overriding Bundle Proration Settings at Customer Level" in *BRM Opcode Guide* for more information.

Configuring Subscription Terms

Subscription terms specify a contract's commitment period, such as 1 month, 1 year, or 2 years. They also specify whether customers:

- Can cancel their contract before the commitment period ends
- Are charged a penalty for canceling early, such as an early termination fee, the contract balance, or a subset of the contract balance
- Have a grace period, during which the penalty is waived
- Have their contracts renewed automatically when the commitment period ends

For example, a term might specify a 1-year commitment period, allow early cancellation with a $250 early termination fee, and renew automatically with the same terms.

When configuring subscription terms, you use the following XML elements:

```
<subscription_terms
penalty_options="contract_balance_and_used_contract_units">
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
   <subscription_terms_name>tempSubscriptionterms</subscription_terms_name>
   <subscription_terms_code>tempSubscriptionterms</subscription_terms_code>
   <description>tempSubscriptionterms</description>
   <renew_flags>1</renew_flags>
   <early_termination_allowed>1</early_termination_allowed>
   <termination_grace_length unit="month">1</termination_grace_length>
   <term_length unit="month">12</term_length>
   <auto_renew_subs_terms_code>tempSubscriptionterms</
auto_renew_subs_terms_code>
   <auto_renew_deal_code>Deal - Subscription Term</auto_renew_deal_code>
   <auto_renew_plan_code>Plan - Subscription Term</auto_renew_plan_code>
   <penalty_amount>100</penalty_amount>
   <resource_id>840</resource_id>
   <revenue_glid>0</revenue_glid>
   <minimum_term_length unit="month">3</minimum_term_length>
</subscription_terms>
```

Table 16-1 describes the XML elements for configuring subscription terms.

Table 16-1 Subscription Term Elements

XML Element Name	Description
subscription_terms_name	The unique name of the term to create. Use the **penalty_options** flag to specify the type of penalty to apply: • **full_charge**: The fee is the full penalty amount. • **used_contract_units**: The fee is reduced in equal amounts over the contract's term. • **contract_balance**: The fee is the balance remaining in the customer's contract. • **contract_balance_and_full_charge**: The fee is the balance remaining in the customer's contract plus the full penalty amount. • **contract_balance_and_used_contract_units**: The fee is the balance remaining in the customer's contract plus the prorated penalty fee.
subscription_terms_code	The unique code for the **/subscription_terms** object.
description	A short description of the subscription term.
renew_flags	A Boolean flag specifying whether to automatically renew the contract: true (**1**) or false (**0**).
early_termination_allowed	A Boolean flag specifying whether customers are permitted to cancel their contract early (**1**) or not (**0**).
termination_grace_length	The length of the grace period, during which the penalty fee is waived. Use the **unit** flag to specify the unit of the grace period: **day**, **month**, **year**.
term_length	The length of the commitment period. For example, for a 6-month commitment period, you would enter 6. Use the **unit** flag to specify the unit of the grace period: **day**, **month**, **year**.
auto_renew_subs_terms_code	The subscription term to use after a contract automatically renews.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 16-1 (Cont.) Subscription Term Elements

XML Element Name	Description
auto_renew_deal_code	The **/deal** to use after a contract automatically renews. This element is required when **renew_flags** is set to **1**.
auto_renew_plan_code	The **/plan** to use after a contract automatically renews. This element is required when **renew_flags** is set to **1**.
penalty_amount	The amount of the early termination fee.
resource_id	The balance element numeric code for the penalty amount, such as 840 for US Dollars.
revenue_glid	The GLID to associate with the penalty amount.
minimum_term_length	The minimum length of a contract's term. If customers cancel their contract early, they must pay the balance of their contract up to the minimum length. Use the **unit** flag to specify the unit of the grace period: **day**, **month**, **year**.

Setting Full Day Proration

To configure full day proration in XML, you use the <**validityRounding**> and <**scaleRounding**> elements.

<**validityRounding**> specifies whether to start the charge offer's validity period at the purchase time or midnight of the purchase day.

Set the <**validityRounding**> element to one of the following:

- **OFF**: Starts at the time of purchase.
- **ON**: Starts at midnight (00:00:00) of the day that the charge offer is purchased.
- **NOT_SET**: Uses your company's systemwide setting.

<**scaleRounding**> specifies how to calculate the scale.

Set the <**scaleRounding**> element to one of the following:

- **ON**: Calculate it based on full days.
- **OFF**: Calculate it based on the <**validityRounding**> setting.
- **NOT_SET**: Use the systemwide setting in the CM pin.conf file.

The following sample XML entries specify to start the validity period at midnight on the purchase day and charge for a full day on the first day of the billing cycle:

```
<?xml version="1.0" encoding="UTF-8"?>
<price_list version="7.2" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="price_list.xsd">
<product date_range_type="EVENT_DATE" partial="no" type="subscription">
    <product_name>MCF - $30</product_name>
    <product_code>MCF30</product_code>
    <offer_validity_rounding>NOT_SET</offer_validity_rounding>
    <scale_rounding>OFF</scale_rounding>
 </price_list>
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

Adding Price Tags to Products and Discounts

You can use price tags to override prices of products and discounts at purchase time. To do so, you add predefined price tags to the balance impacts in your price list XML file.

Price tags are defined in **/config/price_tags** objects. For example, the following object defines two price tags with the same name that apply to different currencies.

```
0 PIN_FLD_PRICE_TAGS  ARRAY [0]
1         PIN_FLD_NAME           STR [0] "PRICETAG1"
1         PIN_FLD_DESCR          STR [0] "Dollar Override"
1         PIN_FLD_RULE_TYPE      STR [0]
"ANY"
1         PIN_FLD_CONSTRAINTS    STR [0] ""
1         PIN_FLD_RESOURCE_ID    INT [0] 840
1         PIN_FLD_RESOURCE_UNIT ENUM [0] 0
1         PIN_FLD_PERMITTED      STR [0] "/service/email; /service/ip"
0 PIN_FLD_PRICE_TAGS  ARRAY [1]
1         PIN_FLD_NAME           STR [0] "PRICETAG1"
1         PIN_FLD_DESCR          STR [0] "Euro Override"
1         PIN_FLD_RULE_TYPE      STR [0]
"LIST"
1         PIN_FLD_CONSTRAINTS    STR [0] "5;10;15;20"
1         PIN_FLD_RESOURCE_ID    INT [0] 978
1         PIN_FLD_RESOURCE_UNIT ENUM [0] 0
1         PIN_FLD_PERMITTED      STR [0] "/service/email; /service/ip"
```

To add price tags to products or discounts, set the **<price_tag>** element under the **<balance_impact>** element to the name of a price tag configuration object.

The following sample XML entries add the **PRICETAG1** price tag to the balance impacts.

```
<?xml version="1.0" encoding="UTF-8"?>
<price_list xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="price_list.xsd" version="7.2">
<rate type="default" prorate_first="prorate" prorate_last="full"
step_type="total_quantity_rated">
   <description>€10 a month</description>
   <step_resource_id>978</step_resource_id>
   <quantity_tier>
      <balance_impact scaled_unit="none" flag="sponsorable discountable
proratable">
         <resource_id>978</resource_id>
         <glid>102</glid>
         <fixed_amount>10.0</fixed_amount>
         <price_tag>PRICETAG1</price_tag>
      </balance_impact>
   </quantity_tier>
</rate>
</price_list>
```

You can use the PCM_OP_SUBSCRIPTION_GET_PRICE_TAGS opcode to get the list of price tags that are valid for a resource, unit of measure, service, or product. See "Getting Information about Price Tags" in *BRM Opcode Guide* for more information about this opcode.

Copyright © 2025, 2026, Oracle and/or its affiliates.

When the price list is loaded into BRM, the price tag is validated by comparing values for the following:

- **<price_tag>** under the balance impact to the list of price tag names in configuration objects.
- **<resource_id>** under the balance impact to PIN_FLD_RESOURCE_ID in configuration objects.
- **scaled_unit** in the balance impact element to PIN_FLD_RESOURCE_UNIT in the configuration object.
- **<permitted>** under the product or discount (not shown in this example) to PIN_FLD_PERMITTED in the confguration object.

In this example, BRM would charge a default fixed amount of €10. At purchase time, you could chose from the values configured for the price tag configuration object for CURRENCYPRICE that matches the resource ID 978 for the balance impact. Possible values are 5, 10, 15, or 20. BRM stores the configured values price tag in **/offering_override_values** objects.

Allowing Customers to Exceed Their Credit Limit

When configuring rates in XML, you can specify what happens to subscriptions when customers exceed their credit limits.

To do so, set **type** in the **<rate>** element to one of the following:

- **normal**: Enforce the credit limit. The subscription succeeds, but customers stay within their credit limit. Prorate the subscription according to the available balance.
- **default**: The subscription succeeds, and all available balance is used. The remaining amount is tracked as a negative account balance to be paid at the next top up for prepaid customers or current bill for postpaid customers.
- **loan**: The subscription succeeds. If the customer is eligible for a loan, all available balance is used and a loan is granted for the remaining amount. If the customer is not eligible for a loan, the subscription fails.
  See "About Loans" in *BRM Configuring and Collecting Payments* for more information about loans.
- **deduct_rental**: The subscription succeeds, and all available balance is used. The remaining amount is recorded as an outstanding amount to be paid at the next top up for prepaid customers or the current bill for postpaid customers.
- **outstanding_amount**: The subscription succeeds without using the available balance. The entire amount is recorded as an outstanding amount to be paid at the next top up for prepaid customers or the current bill for postpaid customers.
- **rental_failure**: Enforce the credit limit. The subscription fails. After retrying the subscription the configured maximum number of times, a notification event is sent to an external system for further processing.
- **auto_renew_cancel**: Enforce the credit limit. The subscription fails. After retrying the subscription the configured maximum number of times, a notification event is sent to an external system for further processing, and the subscription is canceled. It will not automatically renew the next cycle.
- **skip_cycle**: The subscription succeeds without using the available balance. Services will still be available, but billing will be skipped for this cycle. During rating, all balance impacts are dropped, and the unrated quantity is set to 0. The cycle forward date is moved to the next cycle.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- **minimum_amount**: The subscription succeeds if the customer's balance is at least a configured minimum amount. If they have enough balance for the full amount, the full amount is charged. If they have less than the full amount, but more than the minimum amount, the minimum amount is charged. If they have less than the minimum amount, the subscription fails, and the behavior is the same as for **rental_failure**.
  For this option, you can also configure the following:
  - The minimum amount in the **<minimum_scaled_amount>** element. By default, this is set to **0**, which means the subscription fails if the balance is less than the total amount.
  - Whether or not to prorate the service according to the minimum amount in the **prorate_quantity** flag of the **<rate>** element. By default, this is set to **yes**, and the service will be prorated. Set this to **no** to grant the full service for the minimum amount.

  This option is helpful if you want to offer your customers a small amount of their service, such as a few MB of data or minutes as an emergency stop-gap until they have enough balance to afford the full service.

For the **rental_failure**, **auto_renew_cancel**, and **minimum_amount** options, you can also configure the following when defining the product:

- The maximum number of times to retry the subscription before it fails. Configure this in the **<max_retry_count>** element of the product. The default is **0**, which sends the failure notification immediately without retrying.
- The frequency of subsequent retries. Configure this in the charge offer's **<retry_unit>** and **<retry_offset>** elements. The values for **<retry_unit>** are:
  - **1**: Seconds
  - **2**: Minutes
  - **3**: Hours
  - **4**: Days
  - **5**: Months
  - **6**: Years
  - **10**: Weeks

  The default for **<retry_unit>** is 4 and **<retry_offset>** is 1.

The **/purchased_product** object stores the current number of retries for the subscription in the PIN_FLD_RETRY_COUNT field and the date and time of the next retry in the PIN_FLD_NEXT_RETRY_T field.

If you do not configure the retry parameters at the offer level, the value for the **max_retry_count** business parameter is used. By default, this is set to **0**, which fails the subscription immediately without retrying. See "Configuring the Maximum Number of Subscription Retries" in *PDC Creating Product Offerings* for more information about setting this parameter.

The following XML rate and product example specifies charging a minimum amount and prorate the service when customers exceed their credit limit. It also specifies to retry failed subscriptions a maximum of 3 times, every 6 hours.

```
<rate prorate_first="prorate" prorate_last="prorate" prorate_quantity="yes"
step_type="total_quantity_rated" type="minimum_amount">
  <description>Monthly Cycle Forward $12</description>
  <step_resource_id>0</step_resource_id>
  <quantity_tier>
    <balance_impact flag="proratable" scaled_unit="none">
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
      <resource_id>840</resource_id>
      <glid>0</glid>
      <fixed_amount>2.0</fixed_amount>
      <scaled_amount>10.0</scaled_amount>
      <minimum_scaled_amount>5.0</minimum_scaled_amount>
    </balance_impact>
  </quantity_tier>
</rate>
...
<product date_range_type="EVENT_DATE" partial="no" type="subscription">
    <product_name>MCF - $30</product_name>
    <product_code>MCF30</product_code>
    <priority>0.0</priority>
    <description>Monthly cycle forward charges.</description>
    <permitted>/service/ip</permitted>
    <max_retry_count>3</max_retry_count>
    <retry_offset>6</retry_offset>
    <retry_unit>3</retry_unit>
</product>
```

You can also use the **importexportpricing** utility or the PCM_OP_PRICE_SET_PRICE_LIST opcode to update these values.

Transitioning Plans and Deals

When configuring plan or deal transitions in XML, you can specify how to apply charges when customers perform transitions in the middle of their billing cycle. To do so, set **proration_flag** under the **<transition>** element to one of the following:

- **force_prorate**: Prorates charges for both plans or both deals. This is the default.
- **chg_full_from_svc**: Applies the full charges for the original plan or deal, and applies no charges for the new deal or plan.
- **chg_full_to_svc**: Refunds the charges for the original plan or deal, and applies the full charges for the new plan or deal.

The following sample XML configures BRM to prorate charges from both deals when your customers transition from one deal to another in the middle of their billing cycle:

```
<transition fee_flag="3" service_type="/service/ip" type="1"
proration_flag="force_prorate">
   <from type="/deal">Deal A</from>
   <to type="/deal">deal_product_not_aligned_inadvance</to>
</transition>
```

Configuring Add-On Products in Deals

When creating deals in XML, you can add base products and add-on products. All products in deals are base products by default, which means they can be purchased without any prerequisites. Add-on products require customers to own a valid base product.

You customize a product in a deal to be an add-on product by including the **<validity_align_mode>** element under **<deal_product>**. The add-on product's validity start date is the end date of a product that you specify. For example, assume product A has a

Copyright © 2025, 2026, Oracle and/or its affiliates.

validity period from June 1 through June 15. If you specify to align add-on product B's validity period with product A, product B's validity start date would be June 15.

You specify the product on which to align an add-on product's validity dates by setting the **<validity_align_mode>** element to one of the following:

- **base**: Aligns the validity dates with the specified base product. If set to **base**, you must also include these elements:
  - **<base_product_name>** set to the name of the base charge offer on which to set validity dates.
  - **<base_product_code>** set to the code of the base charge offer on which to set validity dates.
- **any_base_earliest**: Aligns the validity dates with the active base charge offer that expires first.
- **any_base_latest**: Aligns the validity dates with the active base charge offer that expires last.
- **any_earliest**: Aligns the validity dates with the active charge offer that expires first.
- **any_latest**: Aligns the validity dates with the active charge offer that expires last.

The following sample XML entries create a deal that includes an add-on product with a validity date aligned with that of **BASE_PRODUCT_SMS**:

```
<deal customization_flag="optional" ondemand_billing="no">
   <deal_name>Dealp1d1</deal_name>
   <description>Dealp1d1</description>
   <permitted>/service/ip</permitted>
   <deal_product status="active">
      <product_name>p1</product_name>
      <product_code>p1</product_code>
      <quantity>1.0</quantity>
      <purchase_start unit="month">5</purchase_start>
      <purchase_discount>0.0</purchase_discount>
      <cycle_start unit="month">5</cycle_start>
      <cycle_discount>0.0</cycle_discount>
      <usage_start unit="month">5</usage_start>
      <usage_discount>0.0</usage_discount>
      <purchase_mode>0</purchase_mode>
      <validity_align_mode>base</validity_align_mode>
      <base_product_name>BASE_PRODUCT_SMS</
base_product_name>
      <base_product_code>abcd-kfhg-sdjd-fgjk</base_product_code>
   </deal_product>
</deal>
```

You can also create or modify a deal with add-on products by using the PCM_OP_PRICE_SET_PRICE_LIST opcode. However, after customers subscribe to your deals, further updates to the product do not affect any purchased products.

You can override the add-on product's validity dates for a specific customer by using the PCM_OP_SUBSCRIPTION_PURCHASE_DEAL or PCM_OP_CUST_COMMIT_CUSTOMER opcode. See "Overriding Add-On Product Validity Dates in a Bundle" in *BRM Opcode Guide* for more information.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Purchasing the Same Product or Discount Multiple Times

When creating deals in XML, you can specify what happens if customers purchase the same product or discount more than once. The additional product or discount can be purchased as a new subscription, as a replacement of the existing subscription, or as an extension of the existing subscription if the additional product is purchased within a specified grace period.

Set the value for the **<purchase_mode>** element under the **<deal_product>** or **<deal_discount>** element to one of the following:

- **0**: The additional product or discount is purchased as a new, unrelated subscription. The balance impacts and validity periods of the old and new subscription are completely independent. This is the default option.
- **1**: The additional product or discount is purchased as an extension to the existing subscription if it is purchased within the specified grace period. The new resources are added to the existing balance group. The validity period for the resource is then set to either the old validity end date or the new validity end date, whichever is later.

  For example, on June 1, a customer purchases a deal that grants 3GB of data, is valid for 7 days, and has a grace period of 4 days. On June 3, after using 1GB, they purchase the deal again. The new 3GB balance is added to the remaining balance, for a total remaining balance of 5GB. BRM compares the validity periods of the old and new product and sets the validity end date for the total balance to the later of the two end dates, June 10.
- **2**: The additional product or discount is purchased as an extension to the existing subscription if it is purchased within the specified grace period. The new resources are added to the existing balance group. The validity period for the resource is then set by adding the new validity period to whatever remains of the old validity period.

  For example, on June 1, a customer purchases a deal that grants 3GB of data, is valid for 7 days, and has a grace period of 4 days. On June 3, after using 1GB, they purchase the deal again. The new 3GB balance is added to the remaining balance, for a total of 5GB. BRM adds the validity period of the new product to the old deal, and sets the validity end date for the total balance to June 15.
- **3**: The additional product or discount replaces the existing subscription. The existing subscription is canceled, any configured proration is applied, and the new subscription is created, with all resources and validity periods set as though for a new purchase.

  For example, on June 1, a customer purchases a deal that grants 3GB of data and is valid for 7 days. On June 3, after using 1GB, they purchase the deal again. The old subscription is canceled, any remaining balance is prorated, and the new 3GB balance is created, with a validity end date of June 10.
- **4**: The additional product or discount is purchased as an extension to the existing subscription if it is purchased within the specified grace period. The new resources are added to a new balance group. The new balance's validity period starts on the purchase date, and ends either on the old validity end date or the new validity end date, whichever is later.

  For example, on June 1, a customer purchases a deal that grants 3GB of data, is valid for 7 days, and has a grace period of 4 days. On June 3, after using 1GB, they purchase the deal again. The new 3GB balance is added to a new balance group, separate from the remaining 2GB in the old balance group.

  The new balance is valid starting from June 3. BRM calculates the validity end date by comparing the validity periods of the old and new balances and setting the validity end

Copyright © 2025, 2026, Oracle and/or its affiliates.

date for the new balance to the later of the two end dates, June 10. The old balance retains the original validity dates.

- **5**: The additional product or discount is purchased as an extension to the existing subscription if it is purchased within the specified grace period. The new resources are added to a new balance group. The new balance's validity period starts on the end date for the old balance's validity and the end is set by adding any remaining validity from the old balance to the total validity of the new balance.

  For example, on June 1, a customer purchases a deal that grants 3GB of data, is valid for 7 days, and has a grace period of 4 days. On June 3, after using 1GB, they purchase the deal again. The new 3GB balance is added to a new balance group, separate from the remaining 2GB in the old balance group.

  The new balance is valid starting June 8. BRM calculates the new balance's validity end date by adding the validity period of the new balance to what remains of the old balance, and setting the validity end date for the new balance to June 15. The old balance retains the original validity dates.

When you extend the existing product or discount by setting **<purchase_mode>** to **1**. **2**, **4**, or **5**, you can also include a grace period during which the additional product or discount must be purchased to qualify for the extension. To do so, set the following elements:

- **<grace_period_offset>**: Set this to the amount of time in the grace period. The default is **0**, which specifies a grace period that never ends.
- **<grace_period_unit>**: Set this to the unit of the grace period: **0** (seconds), **1** (minutes), **2** (hours), or **3** (days). The default is **2** (days).

The following shows a sample of the XML for a product and a discount in a deal with **purchase_mode** set. The sample product specifies to extend the existing subscription and add the new and existing validity periods. The product is extended only if the additional product is purchased within two days of the existing product's expiration date.

The sample discount specifies to replace the existing discount with the new discount.

```
<deal customization_flag="optional" ondemand_billing="no">
          <deal_name>IP Deal</deal_name>
          <description>IP deal with purchase mode</description>
          <permitted>/service/ip</permitted>
          <deal_product status="active">
                    <product_name>Extending Product</product_name>
                    <product_code>ExtendingProduct</product_code>
                    <quantity>1.0</quantity>
                    <purchase_start unit="month">5</purchase_start>
                    <purchase_discount>0.0</purchase_discount>
                    <cycle_start unit="month">5</cycle_start>
                    <cycle_discount>0.0</cycle_discount>
                    <usage_start unit="month">5</usage_start>
                    <usage_discount>0.0</usage_discount>
                    <purchase_mode>2</purchase_mode>
                    <grace_period_offset>2</grace_period_offset>
                    <grace_period_unit>3</grace_period_unit>
                    <renewal_mode>1</renewal_mode>
          </deal_product>
          <deal_discount status="active">
                    <discount_name>Overwriting Discount</discount_name>
              <discount_code>OverwritingDiscount</discount_code>
              <quantity>1.0</quantity>
              <purchase_mode>3</purchase_mode>
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
            </deal_discount>
</deal>
```

Aligning Recurring Charges and Product Validity to a Specific Day of the Month

When creating products in XML, you can use the **calendar_dom** and **cycle_fee_flags** fields to align recurring charges and product validity to a specific day of the month instead of the customer's billing date.

You can determine when to align recurring charges and product validity as follows:

- To align recurring charges and product validity to a specific day of the month, set **calendar_dom** to an integer from 1-31 and set **cycle_fee_flags** to **1**.
- To align recurring charges and product validity to the billing date, set **calendar_dom** to **0**. The value of **cycle_fee_flags** is irrelevant in this scenario.
- To align recurring charges to the billing date but align product validity to a specific day of the month, set **calendar_dom** to an integer from 1-31 and set **cycle_fee_flags** to **0**.

> ⚠ **Caution**
>
> You cannot change the **calendar_dom** value for a product after it has been purchased. If you create a new version of the product with a different value in **calendar_dom**, BRM does not update the value in any purchased products and continues to use the value that was configured for the initial purchase.

For example, to align cycle rates and validity for a product to the 22nd day of the month, the XML input file would include the following:

```
<product type="subscription" partial="no"
date_range_impact_type="INSTANTIATED_DATE">
    <product_name>Product for a Specific Day of the Month</product_name>
    <product_code>5fac608f-ba2b-4e83-adc3-7563e349158f</product_code>
    <priority>1</priority>
    <permitted>/service/telco/gsm/telephony</permitted>
    <event_rating_map tod_mode="start_time" timezone_mode="event"
min_unit="none" incr_unit="none" rounding_rule="nearest"
flag="not_rate_cancelled">
        <event_type>/event/billing/product/fee/cycle/cycle_forward_monthly</
event_type>
        <rum_name>Occurrence</rum_name>
        <min_quantity>0</min_quantity>
        <incr_quantity>1</incr_quantity>
        <rate_plan_name>Rate Plan for a Specific Day of the Month</
rate_plan_name>
        <rate_plan_code>Occurrence-1bb17036-84ef-48be-9458-3b2922cf5276</
rate_plan_code>
        <rate_plan tax_when="never">
            <rate_plan_name>Rate Plan for a Specific Day of the Month</
rate_plan_name>
            <rate_plan_code>Occurrence-1bb17036-84ef-48be-9458-3b2922cf5276</
rate_plan_code>
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
            <currency_id>840</currency_id>
            <event_type>/event/billing/product/fee/cycle/
cycle_forward_monthly</event_type>
            <advance_billing_offset>0</advance_billing_offset>
            <cycle_fee_flags>1</cycle_fee_flags>
            <rate_tier date_range_type="absolute">
                <rate_tier_name>RateTier_0</rate_tier_name>
                <priority>100</priority>
                <date_range relative_start_unit="none"
relative_end_unit="none">

                    ...

                </date_range>
            </rate_tier>
            <rate_tier date_range_type="absolute">
                <rate_tier_name>RateTier_1</rate_tier_name>
                <priority>90</priority>
                <date_range relative_start_unit="none"
relative_end_unit="none">

                    ...

                </date_range>
            </rate_tier>
        </rate_plan>
    </event_rating_map>
    <calendar_dom>22</calendar_dom>
</product>
```

If a customer with a billing date of the 14th of the month purchased the product in this example on January 18, their first cycle forward fee would be applied on January 18, prorated to cover the period between January 18 and January 22 (the **calendar_dom** date.) The next fee would be applied on January 22, at the full amount to cover the period between January 22 and February 22. Bills would be generated on the 14th of each month.

However, if the value of **cycle_fee_flags** in the example above was **0**, the first cycle forward fee would be applied on January 18, prorated to cover the period between January 18 and February 14 (the billing date). The next fee would be applied on February 14, at the full amount to cover the period between February 14 and March 14. In the final month of the product's duration, it would only be valid until the 22nd of the month (the **calendar_dom** date.) If the product was valid for three months, the final fee would be prorated to cover the period from March 14 to March 22.

> **Note**
>
> For products in deals that are suspended and then reactivated, you can specify that the cycle aligns with the reactivation date rather than the original purchase date using the **renewal_mode** element under the **deal_product** element. See "Setting Product Cycle Alignment for Reactivated Deals" for information about how **renewal_mode** interacts with **calendar_dom** and **cycle_fee_flags**.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Splitting Noncurrency Balances into Multiple Validity Periods

When creating products in XML, you can grant noncurrency balances in smaller portions on an incremental basis. For example, you could grant 30 Gigabytes with a one-month validity that is distributed as 1 Gigabyte per day.

Set the **<split_bucket>** element under the **<balance_impact>** element as follows:

```
<split_bucket validity="validityType" unit="unit">amount</split_bucket>
```

where:

- *validityType* is either **bucket** (the balance expires at the end of the incremental validity period) or **total** (the balance expires at the end of the total validity period).
- *amount* and *unit* specify the length of each incremental validity period, such as five hours or two days.

For example, to create an incremental validity period of four hours that has its noncurrency resources expire at the end of the total validity period:

```
<product>
   <event_rating_map>
      <rate_plan>
         <rate_tier>
            <rate>
               <quantity_tier>
                  <balance_impact scaled_unit="none" flag="discountable
grantable sponsorable">
                     <resource_id>100002</
resource_id>
                     <impact_category>default</
impact_category>
                     <glid>0</glid>
                     <fixed_amount>0.0</
fixed_amount>
                     <scaled_amount>-233.0</
scaled_amount>
                     <split_bucket validity="total" unit="hour">4</
split_bucket>
                  </balance_impact>
               </quantity_tier>
            </rate>
         </rate_tier>
      </rate_plan>
   </event_rating_map>
</product>
```

Setting Discount Validity Rounding

When creating discounts in XML, you can specify whether a discount offer's validity period starts when the discount is purchased (such as 15:00) or at midnight (00:00) of the day it was purchased.

Copyright © 2025, 2026, Oracle and/or its affiliates.

To do so, set the **<discount_validity_rounding>** element under the **<discount>** element to one of the following:

- **OFF**: Sets the discount start time to the time of purchase.
- **ON**: Sets the discount start time to midnight (00:00) of the day the discount is purchased.
- **NOT_SET**: Uses the systemwide setting in the **/config/business_profile** object.

You can use the new options with the loadpricelist utility. The following XML example specifies to start the discount validity period at midnight of the day the discount offer is purchased:

```
<?xml version="1.0" encoding="UTF-8"?>
<price_list version="7.2" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="price_list.xsd">
<discount mode="parallel" type="subscription">
<discount_name>Standard GSM Discount</discount_name>
<discount_code>Standard GSM Discount</discount_code>
<priority>0.0</priority>
<permitted>/service/telco/gsm/telephony</permitted>
<discount_usage_validity discount_from_validity_flags="prorate"
discount_partial_validity_flags="prorate"
discount_to_validity_flags="prorate"></discount_usage_validity>
<discount_cycle_validity discount_from_validity_flags="prorate"
discount_partial_validity_flags="prorate"
discount_to_validity_flags="prorate"></discount_cycle_validity>
<discount_usage_map discount_flags="discount_inactivated" snowball_flag="yes">
<event_type>/event/delayed/session/telco/gsm</event_type>
<discount_model>DMFMALL</discount_model>
</discount_usage_map>
<discount_usage_map discount_flags="discount_inactivated" snowball_flag="yes">
<event_type>/event/billing/product/fee/cycle/cycle_forward_monthly</
event_type>
<discount_model>DM10%OFF</discount_model>
</discount_usage_map>
<apply_mode>0</apply_mode>
<discount_validity_rounding>ON</discount_validity_rounding>
</discount>
</price_list>
```

Setting Product Cycle Alignment for Reactivated Deals

When creating deals in XML, you can specify when the cycle should align for products if a customer suspends and then reactivates the deal.

Set the value for the **renewal_mode** element under the **deal_product** element to **1** or **0**. This setting is not valid for discounts.

If **renewal_mode** is set to **0**, then the cycle aligns as described in "Aligning Recurring Charges and Product Validity to a Specific Day of the Month" for more information.

If **renewal_mode** is set to **1**, the following scenarios are possible:

- When **cycle_fee_flags** under **rate_plan** is set to **1**:
  - If **calendar_dom** under **product** is not set, then the cycle aligns with the reactivation date.

Copyright © 2025, 2026, Oracle and/or its affiliates.

  - If **calendar_dom** is set, the cycle aligns with either **calendar_dom** or the reactivation date, whichever is later.
- When **cycle_fee_flags** is set to **0**, the cycle aligns with the billing date.

For a sample of the XML for a product in a deal with **renewal_mode** set to **1**, see "Purchasing the Same Product or Discount Multiple Times" for more information.

You can update the value of **renewal_mode** by using the PCM_OP_PRICE_SET_PRICE_LIST opcode, but after customers subscribe to your deals, further updates to **renewal_mode** have no effect on the purchased products.

Activating Products in Plans and Deals on First Usage

When creating plans or deals in XML, you can specify to activate all products in a plan or deal when one of its products is activated on first usage. To do so, you use the **first_usage** flag in the **<deal>** element (for deals) and in the **<plan>** element (for plans).

Set the **first_usage** flag to one of the following:

- **yes**: Activates all products in a deal or plan upon the first usage of any of its products.

  For example, assume a deal contains products A, B, and C. If the first service a customer uses is in product B, all three products (A, B, and C) are activated at that time.
- **no**: When one product in a deal or plan is activated on first usage, the other products in the plan or deal are not activated.

  For example, assume a plan contains products A, B, and C. If the first service a customer uses is in product B, only product B is activated at that time. This is the default.

The following shows an example of a plan with the **first_usage** flag set:

```
<plan ondemand_billing="no" first_usage="yes">
   <plan_name>Plan01</plan_name>
   <description>Plan01</description>
   <service_deal>
      <service_name>/service/ip</service_name>
      <deal_name>Deal01</deal_name>
      <bal_info_index>0</bal_info_index>
      <subscription_index>0</subscription_index>
   </service_deal>
</plan>
```

Using Date Ranges for Versioning

When creating products in XML, you can add rate plans with new date ranges to create new versions of the same product.

You use the **date_range_type** attribute for the product to specify whether existing subscriptions move to the new charge or continue with the old charge.

BRM determines which rate plan to apply based on the value of **date_range_type** as follows:

- **EVENT_DATE**: The charge event date determines which rate plan is applied. Any existing subscriptions will use the new rate plan after the existing subscription's end date. This is the default behavior.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- **PURCHASE_DATE**: The original purchase date determines which rate plan is applied. Any existing subscriptions will continue to use the old rate plan. Any new subscriptions purchased within the new rate plan's date range will use the new rate plan.

  > **Note**
  >
  > If a customer transfers their subscription as part of a service group transfer and **date_range_type** is set to **PURCHASE_DATE**, the rate plan will be determined based on the service transfer date, not the original purchase date.

- **INSTANTIATED_DATE**: The original service instantiation date determines which rate plan is applied. Any existing subscriptions will continue to use the old rate plan. Any new subscriptions instantiated during the new rate plan's date range will use the new rate plan. This option is useful if your services often have a long delay between purchase and instantiation date. It allows customers with existing subscriptions to continue with the old rate plan, while those who purchase under the old rate plan can take advantage of the new rate plan if it comes into effect before their new service is available.

The following shows an example of a product using **date_range_type**:

```
<?xml version="1.0" encoding="UTF-8"?>
<price_list version="7.2" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="price_list.xsd">
    <product partial="no" type="subscription" date_range_type="EVENT_DATE">
        <product_name>prod1</product_name>
        <product_code>prod1</product_code>
            ...
    </product>
</price_list>
```

The following examples describe how the value of **date_range_type** affects the rate plan applied. In the examples, you have an existing product with a rate plan that credits 1GB of free data every week. You add a new rate plan crediting 1.5GB, set the end date of the old rate plan to June 10, and set the start date for the new rate plan to June 11. Depending on the value of **date_range_type** for the product, the following happens:

- If **dateRangeImpactType** is set to **EVENT_DATE**, all subscriptions get 1GB until June 10 and 1.5GB thereafter.
- If **dateRangeImpactType** is set to **PURCHASE_DATE**, subscriptions purchased before June 10 continue to get 1GB, even for events occurring after June 10. Subscriptions purchased after June 10 get 1.5GB.
- If **dateRangeImpactType** is set to **INSTANTIATED_DATE**, subscriptions instantiated before June 10 continue to get 1GB, even for events occurring after June 10. Subscriptions instantiated after June 10 get 1.5GB, even if they were purchased before.

Defining Product Specification Attributes for Pricing Components

When creating products, discounts, chargeshares, deals, and packages in XML, you can define product specification attributes to store information used by external applications, such as enterprise product catalogs, beyond what is provided in BRM.

You define product specification attributes in **attributes** elements with nested **name** and **value** elements. The following examples show product specification attributes for each pricing component:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Products:

```
<product partial="no" type="subscription">
    <product_name>Product with product specification attributes</
product_name>
    <product_code>Product 1</product_code>
    <priority>0.0</priority>
    <description>cycle_forward_monthly</description>
    <permitted>/service/ip</permitted>
    ...
    <attributes>
        <name>Lifecycle Status</name>
        <value>Test</value>
    </attributes>
    <attributes>
        <name>External Reference ID</name>
        <value>09876</value>
    </attributes>
    ...
</product>
```

- Discounts:

```
<discount mode="parallel" type="subscription">
        <discount_name>Discount with product specification attributes</
discount_name>
        <discount_code>Discount 1</discount_code>
        <priority>0.0</priority>
        <permitted>/service/ip</permitted>
        ...
    <attributes>
                <name>Lifecycle Status</name>
                <value>Suspend</value>
        </attributes>
        <attributes>
                <name>External Reference ID</name>
                <value>98645</value>
    </attributes>
    ...
</discount>
```

- Chargeshares:

```
<sponsorship>
        <sponsorship_name>Standard GSM Charge Share</sponsorship_name>
    ...
        <attributes>
                <name>Lifecycle Status</name>
                <value>Retired</value>
        </attributes>
        <attributes>
                <name>External Reference ID</name>
                <value>24601</value>
        </attributes>
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
    ...
</sponsorship>
```

- Deals:

```
<deal customization_flag="optional" ondemand_billing="no">
    <deal_name>Deal with product specification attributes</deal_name>
        <permitted>/service/ip</permitted>
        <deal_product status="active">
            <product_name>Product with product specification attributes</
product_name>
            <product_code>Product 1</product_code>
            <quantity>1.0</quantity>
        </deal_product>
        <deal_discount status="active">
            <discount_name>Discount with product specification attributes</
discount_name>
            <discount_code>Discount 1</discount_code>
            <quantity>1.0</quantity>
        </deal_discount>
    ...
        <attributes>
                <name>Lifecycle Status</name>
                <value>Draft</value>
        </attributes>
        <attributes>
                <name>External Reference ID</name>
                <value>8675309</value>
    </attributes>
    ...
</deal>
```

- Plans:

```
<plan ondemand_billing="no">
        <plan_name>Plan with product specification attributes</plan_name>
        <service_deal>
            <service_name>/service/ip</service_name>
            <deal_name>Deal with product specification attributes</
deal_name>
        </service_deal>
    ...
        <attributes>
                <name>Lifecycle Status</name>
                <value>Release</value>
        </attributes>
        <attributes>
                <name>External Reference ID</name>
                <value>011235813</value>
        </attributes>
    ...
</plan>
```

Although BRM does not provide validation for these attributes by default, you can configure validations in the PCM_OP_SUBSCRIPTION_POL_VALIDATE_OFFERING opcode. See "Validating Extended Attributes" in *BRM Opcode Guide*.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Storing and Managing Product Specification Attributes

When you import your pricing objects, product specification attributes are stored in an **/offer_attribute_group** object. The POID for the **/offer_attribute_group** object is stored in the **PIN_FLD_ATTRIBUTE_OBJ** field of the pricing object. For example, the flist for a deal with product specification attributes would contain the following:

```
0    PIN_FLD_POID    POID [0] 0.0.0.1 /deal 95359 0
0    PIN_FLD_ATTRIBUTE_OBJ    POID [0] 0.0.0.1 /offer_attribute_group 96383 0
```

You can use the loadpricelist utility to manage product specification attributes as follows:

- Load a price list to add or modify product specification attributes. The values in the **/offer_attribute_group** objects are updated, but the **PIN_FLD_ATTRIBUTE_OBJ** value for the pricing object remains the same.
- Load a price list to remove product specification attributes from a pricing object. The attributes are removed from the **/offer_attribute_group** object. If you remove all product specification attributes from a pricing object, the value of **PIN_FLD_ATTRIBUTE_OBJ** is changed to a null value (**POID [0] 0.0.0.0 0 0**), and the empty **/offer_attribute_group** object remains.
- Download a price list that includes product specification attributes from the database. The utility converts PIN_FLD_ATTRIBUTE_OBJ to **attributes** elements with the appropriate **name** and **value** elements in the output XML.
- Delete a pricing object from the database. Any associated **/offer_attribute_group** objects are also deleted.

Stop Rating Inactive, Canceled, or SuspendedActive Accounts

When configuring cycle rates and purchase rates in XML, you can specify whether BRM should continue charging or stop charging accounts that have the following:

- An **Inactive** status
- A **Cancelled** status
- A **SuspendedActive** custom life cycle state

> **Note**
>
> This option is valid only if your system includes a custom life cycle state named **SuspendedActive**. See "Creating Custom Service Life Cycles" in *BRM Managing Customers*.

For each cycle and purchase rate defined in your pricing XML file, set the **<event_rating_map> flag** element to one of the following:

- **not_rate_inactive**: Specifies not to apply charges to inactive accounts.
- **not_rate_cancelled**: Specifies not to apply charges to canceled accounts.
- **rate_lifecycle_suspend**: Specifies to continue applying charges to suspended accounts.

Copyright © 2025, 2026, Oracle and/or its affiliates.

For example, the following specifies to continue applying charges to suspended accounts:

```
<event_rating_map flag="rate_lifecycle_suspend" incr_unit="none"
min_unit="none" rounding_rule="nearest" timezone_mode="event"
tod_mode="start_time">
```

Setting Loan Thresholds for Plans

When creating credit limits for plans in XML, you can use the following elements to set balance thresholds below which a customer is offered a loan:

- **<thresholds_loan>**: When this percent of the balance is consumed, the customer is offered a loan. The percentage must be in 5% increments.
- **<thresholds_fixed_loan>**: When the balance falls below this fixed number, which must be greater than the credit floor and less than the credit ceiling, the customer is offered a loan.

You can set multiple thresholds within the elements, delimited by a space.

When the customer subscribes to the plan, the thresholds are stored in the **/config/credit_profile** object associated with their balance group. For more information about loan thresholds and what happens when they are breached, see "About Loan Thresholds" in *BRM Configuring and Collecting Payments*.

The following sample XML entries specify several threshold elements. The **<thresholds_loan>** and **<thresholds_fixed_loan>** elements are used for offering loans and the **<thresholds>** and **<thresholds_fixed>** elements are used for different notifications or automatic top-ups:

```
<plan ondemand_billing="no">
      <plan_name>Plan for Loan Thresholds</plan_name>
      <plan_code>123456</plan_code>
      <description />
      <credit_limit>
         <resource_id>840</resource_id>
         <limit>100</limit>
         <thresholds>30 60 90</thresholds>
         <thresholds_fixed>25 50 75</thresholds_fixed>
         <thresholds_loan>35 65 95</thresholds_loan>
         <thresholds_fixed_loan>20 60 80</thresholds_fixed_loan>
         <floor>10</floor>
      </credit_limit>
      ...
   </plan>
```

Enabling Dynamic Credit Floors in Plans

You can set BRM to generate a dynamic credit floor from the granted amounts from the sub-balances that are valid for the current cycle for the resource (for example, minutes). To do so, you use the **dynamic_floor** flag in the **<credit_limit>** element.

Set the **dynamic_floor** flag to one of the following:

- **1**: Sets the credit floor to be based on the granted amounts from the sub-balances that are valid for the current cycle for the resource (for example, minutes).

Copyright © 2025, 2026, Oracle and/or its affiliates.

For example, assume a customer has a usual monthly grant of 10GB of data, but for a single charge period has purchased a booster pack of 5 GB of data. The dynamic credit floor updates the credit floor to 15 GB for the current cycle, and then reverts to 10 GB at the start of the next cycle.

- **0**: Does not set a dynamic credit floor. In this case, the fixed credit floor will be used. This is the default.

The following shows an example of a plan with the **dynamic_floor** flag set:

```
<plan ondemand_billing="no">
    <plan_name>Plan for Loan Thresholds</plan_name>
    <plan_code>123456</plan_code>
    <description />
    <credit_limit>
        <resource_id>840</resource_id>
        <limit>100.0</limit>
        <thresholds>50 60</thresholds>
        <thresholds_fixed>80</thresholds_fixed>
        <thresholds_loan>20 40</thresholds_loan>
        <thresholds_fixed_loan>40</thresholds_fixed_loan>
        <floor>0.0</floor>
        <dynamic_floor>1</dynamic_floor>
    </credit_limit>
    ...
 </plan>
```

Creating Tax Selectors for Products

You can create tax selectors for products so that defined tax codes can be applied based on attributes in the **/account**, **/service**, **/event**, or **/profile** objects.

To do so, you define the selector using the **<tax_selector>** element and then set the name of the selector in the **<tax_selector_name>** element under the **<product>** element.

The elements under the **<tax_selector>** element are:

- **selector_name**: Text containing the name of the selector.
- **selector_rules**: Container for a set of selection criteria.
- **criteria**: The selection criteria for the tax code. This has an attribute, **operator**, which specifies the comparison type. It must be either **equal** or **in**. The operator is case-sensitive, and the default is **equal**.
- **object_class**: The name of the class containing the element to be compared.
- **elem_name**: The specific child element of the object in the **<object_class>** element that contains the field to be compared. This can contain multiple layers, separated by periods, for example, **SUB_BAL_IMPACTS.SUB_BALANCES**.
- **field_name**: The name of the field containing the data to be compared.
- **field_value**: The value that is to be compared with the contents of the field specified by **<field_name>**.
- **code**: The tax code that should be mapped if the comparison is successful.

The following shows an example of **<tax_selector>** configuration:

```
<tax_selector>
  <selector_name>tax1</selector_name>
  <selector_rules>
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
    <criteria operator="equal">
      <object_class>/account</object_class>
      <elem_name> PIN_FLD_NAMEINFO </elem_name>
      <field_name>PIN_FLD_CITY</field_name>
      <field_value>Bangalore</field_value>
    </criteria>
    <criteria operator="equal">
      <object_class>/event</object_class>
      <field_name>PIN_FLD_NAME</field_name>
      <field_value>cycle_fwd</field_value>
    </criteria>
    <code>VAT</code>
  </selector_rules>
  <selector_rules>
    <criteria operator="equal">
      <object_class>/account</object_class>
      <elem_name> PIN_FLD_NAMEINFO </elem_name>
      <field_name>PIN_FLD_CITY</field_name>
      <field_value>VSKP</field_value>
    </criteria>
    <criteria operator="equal">
       <object_class>/event</object_class>
       <field_name>PIN_FLD_NAME</field_name>
       <field_value>cycle_arrear</field_value>
    </criteria>
    <tax_code>VAT1</tax_code>
  </selector_rules>
</tax_selector>
```

Following is an example of a product configuration containing the **<tax_selector_name>** element:

```
<product date_range_type="EVENT_DATE" partial="no" type="subscription">
. . .
  <event_rating_map incr_unit="none" min_unit="none" rounding_rule="nearest"
timezone_mode="event" tod_mode="start_time">
. . .
    <rate_plan advance_billing_unit="day" tax_when="now">
      <rate_plan_name>charge2$-session</rate_plan_name>
      <currency_id>840</currency_id>
      <event_type>/event/session</event_type>
      <tax_selector_name>tax1</tax_selector_name>
      <advance_billing_offset>0</advance_billing_offset>
      <cycle_fee_flags>0</cycle_fee_flags>
. . .
    </rate_plan>
. . .
  </event_rating_map>
</product>
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

Creating Tax Exemption Selectors for Products

You can create tax exemption selectors for products so that defined tax exemptions can be applied based on attributes in the **/account**, **/service**, **/event**, or **/profile** objects. For more information about tax exemptions, see "About Tax Exemptions" in *BRM Calculating Taxes*.

To do so, you define the selector using the **<tax_exemption_selector>** element and then set the name of the selector in the **<selector_name>** element under the **<product>** element.

The elements under the **<tax_exemption_selector>** element are:

- **selector_name**: Text containing the name of the selector.
- **selector_rules**: Container for a set of selection criteria.
- **criteria**: The selection criteria for the tax exemption. This has an attribute, **operator**, which specifies the comparison type. It must be either **equal** or **in**. The operator is case-sensitive, and the default is **equal**.
- **object_class**: The name of the class containing the element to be compared.
- **elem_name**: The specific child element of the object in the **<object_class>** element that contains the field to be compared. This can contain multiple layers, separated by periods, for example, **SUB_BAL_IMPACTS.SUB_BALANCES**.
- **field_name**: The name of the field containing the data to be compared.
- **field_value**: The value that is to be compared with the contents of the field specified by **<field_name>**.
- **code**: The exemption code that should be mapped if the comparison is successful.

The following shows an example of **<tax_exemption_selector>** configuration:

```
<tax_exemption_selector>
  <selector_name>tax_exempt</selector_name>
  <selector_rules>
    <criteria operator="equal">
      <object_class>/account</object_class>
      <elem_name> PIN_FLD_NAMEINFO </elem_name>
      <field_name>PIN_FLD_CITY</field_name>
      <field_value>Bangalore</field_value>
    </criteria>
    <criteria operator="equal">
      <object_class>/event</object_class>
      <field_name>PIN_FLD_NAME</field_name>
      <field_value>cycle_fwd</field_value>
    </criteria>
    <code>student</code>
  </selector_rules>
  <selector_rules>
    <criteria operator="equal">
      <object_class>/account</object_class>
      <elem_name> PIN_FLD_NAMEINFO </elem_name>
      <field_name>PIN_FLD_CITY</field_name>
      <field_value>VSKP</field_value>
    </criteria>
    <criteria operator="equal">
      <object_class>/event</object_class>
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
      <field_name>PIN_FLD_NAME</field_name>
      <field_value>cycle_arrear</field_value>
    </criteria>
    <code>employee</code>
  </selector_rules>
</tax_exemption_selector>
```

Following is an example of a product configuration containing the **<selector_name>** element:

```
<product partial="no" type="subscription">
  <product_name>tax_prod_RPS</product_name>
  <product_code>tax_prod_RPS</product_code>
  <priority>0.0</priority>
  <description>Sample product</description>
  <permitted>/service/ip</permitted>
  <tax_supplier>TS1</tax_supplier>
  <selector_name>tax_exempt</selector_name>
. . .
</product>
```

Importing Tied Discount Offer Configurations into BRM

You can use the **loadpricelist** utility to commit discount offer configurations, including the eligibility specified by the **<tied_discount>** element, from your price list XML into the BRM database. This ensures that BRM recognizes which discount offers are eligible to be tied with specific charge offers at runtime. For more information, see "Setting Discount Offers as Eligible for Tying" in *PDC Creating Product Offerings*. This utility populates the PIN_FLD_TIED_DISCOUNT field in BRM.

Copyright © 2025, 2026, Oracle and/or its affiliates.

17

Real-Time Rating Based on Multiple RUMs

This chapter shows how to use Oracle Communications Billing and Revenue Management (BRM) to set up products to rate events based on multiple ratable usage metrics (RUMs) if you are using Pricing Center.

Topics in this document:

- About Rating Based on Multiple RUMs
- Applying Multiple RUMs to Rate an Event
- Example of Mapping an Event to Multiple RUMs

About Rating Based on Multiple RUMs

By default, BRM rates an event by using a single RUM that is specified in the product's usage map in Pricing Center. You can also set up your products to rate an event based on multiple RUMs. For example, you can set up a Session Event with both Occurrence and Duration selected as RUMs by which the event is measured.

To rate an event using multiple RUMs, you map the event to the RUMs in the product's usage map.

To apply the RUMs, you map the event to the RUMs and the rate plan in the product's usage map. For example, you can map two IP Dialup events to the RUMs, **Duration** and **Occurrence**, in a usage map for an Internet Access product. You can select both the events for rating.

For each event to be rated, click the check box in the ***** column, located between the event number column and the **Event** column.

> **Note**
>
> Specifying multiple RUMS for the same event type and selecting events to be rated are two separate functions, each handled in the Event Map.

When an event is rated using multiple RUMs, a balance impact is applied for each RUM per each resource. For example, if two resources are impacted, two separate balance impacts are created.

For information about setting up products with multiple RUMs, see "Applying Multiple RUMs to Rate an Event".

> **Note**
>
> BRM does not support multiple RUM rating on cycle and purchase fee events.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Applying Multiple RUMs to Rate an Event

To rate an event by using multiple RUMs, you map the event to the RUMs in the product's usage map.

To use multiple RUMs to rate an event, you do the following:

1. Define the RUMs available for rating. See "About Ratable Usage Metrics".
2. Map the event to the RUMs in the product's usage map. See "Example of Mapping an Event to Multiple RUMs".

Example of Mapping an Event to Multiple RUMs

When you create a product, you specify the events to rate for the service, and the rate plans that determine how to charge for those events. To rate an event by using multiple RUMs, you associate the event with the RUMs and the rate plans.

You can map the same event to multiple RUMs by using different rate plans. For example, you can specify a rate plan to charge for a fax event based on the number of pages faxed and another rate plan to charge based on the duration of the fax session.

To map an event to RUMs:

1. Specify the RUMs in the product's usage map.
2. Indicate the events to be rated by clicking the check box in the ***** column, located between the event number column and the event type column.
3. When you finish creating products, deals, and plans, commit your price list to the BRM database.

Copyright © 2025, 2026, Oracle and/or its affiliates.

18

Rating Based on Multiple RUMs with Pipeline Manager

This chapter shows how to use Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager to set up charge offers or rate plans to rate batch events based on multiple ratable usage metrics (RUMs).

Topics in this document:

- About Rating Based on Multiple RUMs with Pipeline Manager
- Applying Multiple RUMs to Rate a Batch Event

About Rating Based on Multiple RUMs with Pipeline Manager

To rate an event using multiple RUMs with Pipeline Manager, you set up multiple Pricing steps. In other words, you configure pipeline rating within the individual pricing.

For each RUM, the data is set in the Event Data Record (EDR) before it is sent to the batch pipeline. Pipeline Manager then uses this data to do the rating based on each RUM. Each RUM is fully rated before rating the next RUM.

For more information about applying multiple RUMs to rate a batch event, see "Applying Multiple RUMs to Rate a Batch Event".

> **Note**
>
> BRM does not support multiple RUM rating on cycle and purchase fee events.

Applying Multiple RUMs to Rate a Batch Event

To rate a batch event by using multiple RUMs, you set up multiple Pricing steps.

To use multiple RUMs to rate a batch event, do the following:

1. Define the RUMs available for rating.
2. Set up pricing steps. The following example shows a multi-RUM charge model created for a standard GPRS delayed event:
   - Duration RUM: 0.02 EUR/min
   - Volume Sending RUM: 2 EUR/MB
   - Volume Receiving RUM: 2 EUR/MB

   In this case, you set up the following pricing steps:
   - Configure the pricing for the **Duration RUM** as follows:
     - Beat: 60
     - Charge: 0.02

Copyright © 2025, 2026, Oracle and/or its affiliates.

  - Charge Base: 60
- Configure the pricing for the **Volume Receiving RUM** as follows:
  - Beat: 2048
  - Charge: 2.00
  - Charge Base: 1048576
- Configure the pricing for the **Volume Sending RUM** as follows:
  - Beat: 2048
  - Charge: 2.00
  - Charge Base: 1048576

Copyright © 2025, 2026, Oracle and/or its affiliates.

19

Real-Time Rating Based on Date and Time

This chapter explains how to use Oracle Communications Billing and Revenue Management (BRM) to set up rates based on date and time for real-time rating.

Topics in this document:

- About Rating Based on Date and Time
- Ways to Specify When Rates Are Valid
- Using Rate Tiers to Set Up Rating Based on Date and Time
- Using the Correct Time Zone
- Using Event Start Time to Determine a Product's Validity
- Dynamically Changing One-Time and Recurring Fees Based On the Date

About Rating Based on Date and Time

BRM rates billable events by capturing data about the events (for more information, see "About Creating a Price List"), such as the date and time each event occurred. Because BRM is aware of the date and time of an event, the rating opcodes can use the date and time to determine how to rate the event.

Ways to Specify When Rates Are Valid

You can specify when rates are valid in the following ways:

- **In products.** You can specify when a product is valid, including all the rates it contains. See "Restricting When a Product Is Available".
- **In rate tiers.** You can use rate tiers to specify a set of rates, each based on a specific time or day. See "Using Rate Tiers to Set Up Rating Based on Date and Time".

When you create rates based on date and time, you can specify how to recognize the time that the event occurred (for example, at the start, end, or duration of the event).

You specify a time-of-day mode to indicate when a rate should be applied relative to either an event's start time, its end time, or the difference between the two. This is especially useful when an event overlaps a time boundary where a rate changes.

For example, 12 a.m. Monday might be the boundary between a special weekend rate and a regular weekday rate:

- To apply the special weekend rate to the entire event, specify **Start Time**.
- To apply the regular weekday rate to the entire event, specify **End Time**.
- To apply the weekend rate to the portion of the event before midnight and the weekday rate after midnight, specify **Timed**.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> Midnight (12 a.m.) is shown as 00.00.00.

Using Rate Tiers to Set Up Rating Based on Date and Time

You use rate tiers to set up valid dates and times for rates. You can include multiple rate tiers in a rate plan.

Specifying Which Rate Tiers Are Applied First

As you create rate tiers, they appear on a list. BRM applies rates from the top of the list down. You can change the order in which rate tiers are applied by changing their places in the list.

For example, at the top of the rate tier list you could insert a promotional rate tier valid only until the end of May 2003. BRM applies that rate only until May 31, 2003. After that, the promotional rate tier is no longer valid, and BRM no longer applies it.

Specifying How Long Rate Tiers Are Valid

You specify how long a rate tier is valid by specifying a duration. You can specify that a rate tier is always valid or is valid for either an absolute or a relative date range.

For example, your basic IP Fax rate tier should not change and always applies. You could specify an absolute date range of 12/1/01 through 12/31/01 for a promotional holiday rate tier. You could also specify a relative date range of 30 days from purchase until 60 days from purchase to give your customers special rates during the second month.

Specifying When Rate Tiers Are Valid

You specify when a rate tier is valid by selecting ranges of dates, days of the week, and times of day. For example:

- For each rate tier, you specify a valid date range. For example, you can specify that a rate tier is valid for the entire year 2001.
- For each date range, you specify one or more valid day-of-the-week ranges. For example, you can specify that a rate tier is valid Monday through Friday.
- For each day-of-the-week range, you specify one or more valid time-of-day ranges. For example, you can specify that a rate tier is valid between 6 a.m. and 8 p.m.

Using Date, Day, and Time Ranges Together

Date ranges consist of a start date and an end date between which a rate tier is valid. For example, a promotional rate tier for usage fees is valid only for the month of January.

Day ranges consist of a list of days during which a rate is valid. For example, a rate tier for off-peak IP usage fees is valid only on weekends.

Time-of-day ranges consist of a start time and an end time between which a rate tier is valid. For example, an after-hours rate tier for IP usage fees is valid only between 10 p.m. and 6 a.m.

Within each date range, you can use multiple day ranges; within each day range, you can use multiple time-of-day ranges.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Using the Correct Time Zone

Because BRM uses the date and time of an event to determine how to rate the event, using the correct time zone during rating is important. You specify whether real-time rating uses the *event* time zone or the *server* time zone to rate the event. By using time zone data, BRM can correctly calculate event start and end times before rating occurs.

Specifying a Time Zone Mode

When you create a product in Pricing Center, you specify which of the following time zone modes BRM uses for rating events whose rates are based on time of day:

- **Event:** BRM uses the time zone in which the event occurs. For example, if a user in California logs on to a terminal server located in California, BRM uses California time, even if the BRM server that rates the event is in New York. This is the default time zone mode.
- **Server:** BRM uses the time zone in which the BRM server that rates the event is located. For example, if a California user logs on to a terminal server located in California but the BRM server that rates the event is in New York, BRM uses New York time. For information about setting your system's server time zone, see "Resetting the Server Time Zone".

For information about setting the time zone mode, see the Pricing Center Help.

Resetting the Server Time Zone

The server time zone is where the server you use for rating resides.

By default, the server time zone is set to **America/Los_Angeles**. If your rating server is not in that time zone, reset the server time zone to the time zone ID indicating the server location.

> **Note**
>
> BRM supports only the time zone IDs defined in instances of the Java TimeZone class.

To reset the server time zone:

1. Get a list of time zone IDs in the Java TimeZone class by running the following function:

   ```
   TimeZone.getAvailableIDs()
   ```

2. In the list, find the ID for the time zone in which your server resides and write it down.
3. Open the *BRM_home***/sys/cm/timezones.txt** file, where *BRM_home* is the directory in which BRM is installed.

   This file lists all the time zone IDs available for use in your system.
4. If the ID for the time zone in which your server resides is *not* in the **timezones.txt** file, do the following:
   a. Open the *BRM_home***/sys/cm/sample_timezone.txt** file.
   b. Add an entry to the **sample_timezone.txt** file for the time zone in which your server resides by following the instructions in the file.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> When configuring time zone support in BRM, note that:
>
> - BRM supports only the time zone IDs defined in instances of the Java TimeZone class.
> - Each time zone ID entry in the **sample_timezone.txt** file contains 13 mandatory parameters. The value of the **tzone_name** parameter must match the name of a time zone ID in the Java TimeZone class. To get a list of time zone IDs in the TimeZone class, run the `TimeZone.getAvailableIDs()` function.
> - To add multiple time zone IDs to the file, enter each time zone ID entry on a separate line.

c. Save and close the **sample_timezone.txt** file.

5. Open the *BRM_home*/**sys/cm/pin.conf** file.
6. If you added an entry to the **sample_timezone.txt** file in step 4, add the following entry to the **pin.conf** file:

   ```
   - fm_rate timezone_file sample_timezone.txt
   ```

7. Find the following entry in the **pin.conf** file:

   ```
   -rating_timezone server_timezone_ID
   ```

   where *server_timezone_ID* specifies the ID of the time zone in which the BRM server performing rating is located.

8. Replace the current *server_timezone_ID* value with the time zone ID of your BRM rating server.

   For example, if your rating server is in Saskatchewan, use the following time zone ID:

   ```
   - fm_rate rating_timezone Canada/Saskatchewan
   ```

   > **Note**
   >
   > To add multiple time zones to the file, add separate **-fm_rate rating_timezone** entries.

9. Save and close the file.
10. Stop and restart the CM.

Configuring Applications to Generate Time Zone Data

When a BRM client application or service integration client (such as RADIUS Manager and Universal Event Loader) generates an event, that event includes time zone data. You must configure event-generating applications to generate the correct time zone data. This means creating an entry in a **pin.conf** or properties file. See the configuration section of your client application or the service integration component documentation for details.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Using Event Start Time to Determine a Product's Validity

By default, in real-time rating an event's end time is used to determine a product's validity.

You can specify that the event start time be used for determining the product's validity by adding an entry to the CM configuration file.

To use the event start time to determine the product's validity:

1. Open the *BRM_home***/sys/cm/pin.conf** file.
2. Add the following entry:

   ```
   - fm_rate use_prods_based_on_start_t /event
   ```

   You can use **/event** to use the start time for all events, or use a subclass of **/event** to specify just events of a given type.
3. Stop and restart the CM.

Dynamically Changing One-Time and Recurring Fees Based On the Date

You can configure BRM to dynamically change the rate for one-time and recurring fees per customer based on the date. For example, when a customer cancels a product, you could charge a default amount of $50, but charge $45 in June and $40 in July.

You set up BRM to dynamically change one-time and recurring fees by doing the following:

- Creating a price tag configuration object associated with a particular service, resource, or unit and load it using the **load_config** utility. See "load_config" in *BRM Developer's Guide* for more information about this utility.
  For example, create a price tag configuration object named PRICETAG1 that applies to the **/service/ip** service and resource ID 840 (USD), with allowed prices in a range between $10-$30.
- Adding the price tag to the balance impacts for a product using the **loadpricelist** utility. See "Adding Price Tags to Products and Discounts".
  If you are using a third-party client to call the PCM_OP_PRICE_SET_PRICE_LIST opcode, add the price tags by setting the PIN_FLD_PRICE_TAG field under the balance impacts to the name of the appropriate price tag. See "Committing Price List Data to the BRM Database" for more information about PCM_OP_PRICE_SET_PRICE_LIST.

  You can use the PCM_OP_SUBSCRIPTION_GET_PRICE_TAGS opcode to get the list of price tags that are valid for a resource, unit of measure, service, or product. See "XREF" for more information about this opcode.
- Adding date ranges and specifying the price tag name, amount, and resource ID when creating the price override using the PCM_OP_SUBSCRIPTION_CRUD_OFFER_OVERRIDE opcode. See "Overriding Charges and Discounts for a Period of Time" in *BRM Opcode Guide*.
  For example, add a date range from January 1, 2032 to February 1, 2032 that includes PRICETAG1 with the amount set to $10, and another date range from February 2, 2032 to March 1, 2032 that includes PRICETAG1 with the amount set to $20.

When rating events, BRM determines whether the price tag in a balance impact matches a price tag and date range combination defined in the **/offering_override_values** object. If it

Copyright © 2025, 2026, Oracle and/or its affiliates.

does, BRM charges the rate associated with the price tag. If it does not find a match, BRM charges the default rate.

After your pricing tag and date range combinations expire, you can remove them from an **/offering_override_values** object by running the **pin_clean_offer_override** utility. The utility removes pricing tag and date range combinations by event type or date. For more information, see "pin_clean_offer_override".

Copyright © 2025, 2026, Oracle and/or its affiliates.

20

Rating by Date and Time with Pipeline Manager

This chapter describes how to set up time models for the Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager.

Topics in this document:

- About Rating by Date and Time with Pipeline Manager
- Configuring Date and Time Rating
- About Special Day Rates

About Rating by Date and Time with Pipeline Manager

A time model defines a combination of time periods. Each time period covers a specific time range for one or more days.

You use time models and time periods to charge different prices for the same service depending on the day and time.

Each time period contains the following information:

- **Day code**: The day or days the time period covers. For example, **Weekdays (Mon.-Fri.)**.
- **Time interval**: The time range the time period covers for the day code. For example, **08:00-17:00 Peak Time**.

> **Note**
>
> A time model must cover all the days of the week. Within a time model, although you can create multiple day codes, you cannot use the same day in multiple day codes.

For example, one time model contains these time periods and is associated with pricing that charges for GSM telephony service as shown in Table 20-1:

Table 20-1 Rating by Date and Time

Time Period	Day Code and Time Interval	Charge from Pricing (Euro Cents per Minute)
WEEKPEAK - Weekdays Peak	Weekdays (Mon.-Fri.) 08:00 - 17:00	.15
WEEKOFF1 - Weekdays Off-Peak 1	Weekdays (Mon.-Fri.) 17:00 - 22:00	.12
WEEKOFF2 - Weekdays Off-Peak 2	Weekdays (Mon.-Fri.) 22:00 - 08:00	.10

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 20-1 (Cont.) Rating by Date and Time

Time Period	Day Code and Time Interval	Charge from Pricing (Euro Cents per Minute)
WENDOFF - Weekend Off-Peak	Weekends (Sat., Sun., Holidays) 00:00-24:00	.09

> **Note**
>
> Special days are not treated as normal days, so you can add special days to a time period without conflicting with any other days.

You include time models and time periods in charge configurations, which specify a pricing for a charge version. When a call is rated, the time model and time period, with the service code and impact category, determine the charge.

About Special Day Calendars

A special day calendar is a set of dates that you designate as being eligible for special rating. Each date can either be a specific date valid only in one year, such as May 11, 2003 for Mother's Day in the US, or a recurring date valid each year, such as January 1 for New Year's Day.

You include a calendar in a charge. The rating module uses the calendar to determine if a date in an event detail record (EDR) is a special day. You can include the same calendar in multiple charges, but you can also create different calendars for different charges.

Configuring Date and Time Rating

To configure date and time rating, do the following:

1. Create time models in Pricing Center or PDC and link them to charge configurations.
2. Create special day calendars in Pricing Center or PDC and link them to charges.
3. Configure the DAT_TimeModel and DAT_Calendar modules.

> **Note**
>
> When you update time model and calendar data, you must use the **Reload** semaphore to reload data into the DAT_TimeModel and DAT_Calendar modules.

About Special Day Rates

Use special day rates to create special rates for specific times or days, independent of time models. You can create two types of special day rates:

- **Global**: Special day rates that apply to all accounts in your system. See "Setting Up Global Special Day Rates".

Copyright © 2025, 2026, Oracle and/or its affiliates.

- **Account-specific**: Special day rates that apply only to one or more specific accounts in your system. You must create an extended rating attribute (ERA) for each account for this type of special day rate.

  See "Setting Up Account-Specific Special Day Rates".

> **Note**
>
> When you create a special day discount, such as a birthday discount, any usage that begins on the discounted day and continues into the next, non-discounted day is fully discounted. Discounting does not split the charge packet at midnight but applies the discount for the entire call. For example, a customer receives a discount for 10% off on all birthday calls. On his birthday, the customer makes a 40 minute call that starts at 11:50 p.m. and ends on 12:30 a.m. the next day. The entire 40 minutes is discounted 10%.

Setting Up Global Special Day Rates

To set up special day rates that apply to all accounts, do the following:

1. Define special day rates in Pricing Center or PDC.
2. Configure the FCT_Dayrate module to calculate charges for date-based rates.
3. Configure the DAT_Dayrate module that stores the data used by the FCT_Dayrate module.

> **Note**
>
> When you update special day rate data, you must use the **Reload** semaphore to reload data into the DAT_Dayratemodule.

Setting Up Account-Specific Special Day Rates

To set up special day rates for a single account or group of accounts:

1. Create an ERA for the special day.

   On the **Promotion** tab, select Special Day and enter the name BIRTHDAY and a date. For example, when creating a special day rate for a birthday, enter the date of the subscriber's birthday.

2. In Pricing Center or PCC, create a usage type that corresponds to the special day.

   For example, for a birthday special rate, you might create a usage type with a code of BIRTH.

3. In Pricing Center or PDC, include the usage you defined in a discount.

   You specify the usage type as part of the discount filter. The discount also specifies the amount of the discount for this rate.

4. Configure the IRL_UsageType iRule to map this usage type to EDR data.

   For example, for the usage type BIRTH, add a line similar to the following to the **IRL_UsageType.data** file:

   ```
   N;N;.;.;.;.;.;.;.;.;.;Y;.;.;BIRTH
   ```


Copyright © 2025, 2026, Oracle and/or its affiliates.

21

Real-Time Rating Based on Event or Resource Quantity

This chapter describes how to set up quantity-based rating in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

- About Quantity-Based Rating
- Example 1: Rating Based on the Quantity of an Event
- Example 2: Rating Based on Total Event Quantities
- Example 3: Rating Based on Total Number of Events
- Example 4: Rating Based on a Resource Balance Not Affected by the Rated Event
- Example 5: Rating an Event As If Using a Single Balance Impact Group

Before you read this chapter you should be familiar with rating and price lists. See "About Creating a Price List".

About Quantity-Based Rating

You use quantity-based rating when you must rate events based on previous rating or on the current balance of a resource. For example:

- $1 per fax for the first 10 faxes.
- 50 cents per fax for the next 90 faxes, up to and including the 100th fax.
- Free faxes for 101 or more.

To set up quantity-based rating, you group together balance impacts into *quantity discount brackets*, which are applied for a range of event quantities. For each range, you apply a different bracket group of balance impacts and you charge customers differently for each event quantity.

> **Note**
>
> Quantity-based rating applies only to events rated in real time. You can use discounts to achieve many of the same results for both events rated in real time and events rated in batch by the pipeline.

Selecting a Quantity Discount Bracket Based on Previous Rating

You can use the event quantity as the basis for selecting a quantity discount bracket by using the Continuous option or the Rate Dependent option. When BRM requires only one rate for an event, the Continuous or Rate Dependent options do the same thing. When BRM requires

Copyright © 2025, 2026, Oracle and/or its affiliates.

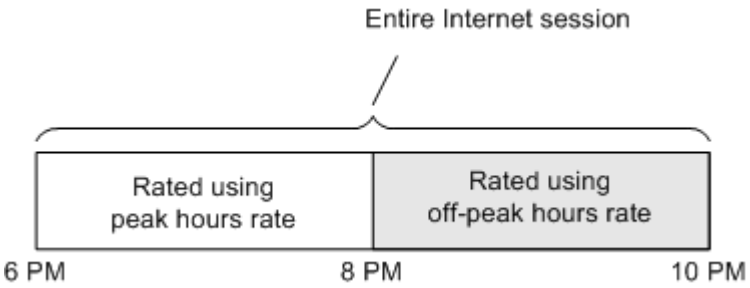
more than one rate for a given event, the difference between Continuous and Rate Dependent becomes important. For example, see the rates for Internet access shown in Table 21-1:

Table 21-1 Connection Times and Discount Calculations

Peak Hours Rate: 6 a.m. - 8 p.m.	Off-Peak Hours Rate: 8 p.m. - 6 a.m.
Quantity Discount Bracket 1: No min-No max Balance Impact: $1 per hour	Quantity Discount Bracket 1: 0-2 hours (first two hours) Balance Impact: $0.50 per hour
Quantity Discount Bracket 1: No min-No max Balance Impact: $1 per hour	Quantity Discount Bracket 2: 2+ hours (subsequent hours) Balance Impact: $0.25 per hour

In the example shown in Table 21-1, your customer connects to the Internet at 6 p.m. and disconnects at 10 p.m. The 2 hours between 6 p.m. and 8 p.m. are rated by the peak hours rate. The 2 hours between 8 p.m. and 10 p.m are rated by the off-peak hours rate. Figure 21-1 shows a timeline for this example.

Figure 21-1 Internet Session Spanning Peak and Off-Peak Rates

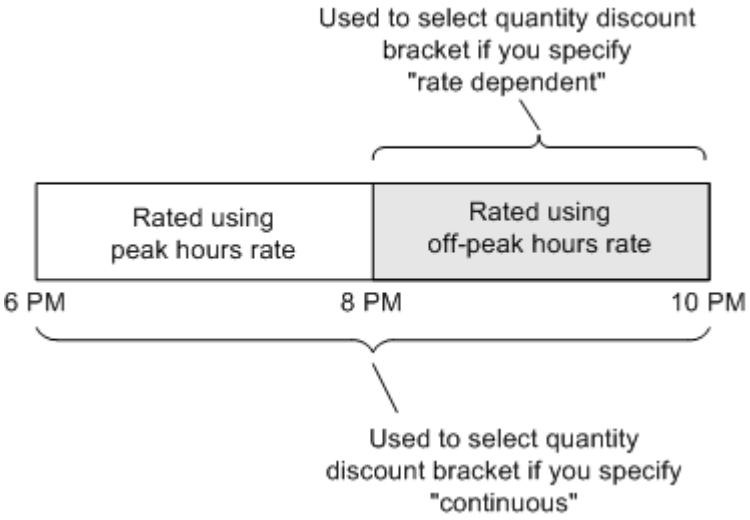


Which quantity discount bracket from the off-peak hours rate is used depends on whether you specify *Continuous* or *Rate Dependent* as shown in Figure 21-2.

- Specify *Continuous* if the quantity used to select the quantity discount bracket for the off-peak hours rate will include *both* the quantity rate by using the peak hours rate and the quantity rated by using the off-peak hours rate.
- Specify *Rate Dependent* if the quantity used to select the quantity discount bracket for the off-peak hours rate will include *only* the quantity rated by using the off-peak hours rate.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 21-2 Continuous Versus Rate Dependent Discounting



> **Note**
>
> The **Resource Balance** option isn't applicable because you aren't basing the charges on a resource balance in the customer's account.

Selecting a Quantity Discount Bracket Based on a Resource Balance

When you specify *Resource Balance*, BRM selects a quantity discount bracket based on the balance of a resource. For example, by using the following rates in Table 21-2 to charge your customers for IP telephony:

Table 21-2 Rating Customers for IP telephony

Peak Hours Rate: 6 a.m. - 8 p.m.	Off-Peak Hours Rate: 8 p.m. - 6 a.m.
Quantity Discount Bracket 1: 0-10 hours of Internet time Balance Impact: $0.10 per minute	Quantity Discount Bracket 1: 0-10 hours of Internet time Balance Impact: $0.05 per
Quantity Discount Bracket 2: 10+ hours of Internet time Balance Impact: $0.07 per minute	Quantity Discount Bracket 2: 10+ hours of Internet time Balance Impact: $0.02 per hour

Between the hours of 6 a.m. and 8 p.m., the peak hours rate is used. If your customer has accumulated fewer than 10 hours of Internet time, your customer is charged $0.10 per minute for IP phone calls. If your customer has accumulated more than 10 hours of Internet time, your customer is charged $0.07 per minute for IP phone calls.

Between the hours of 8 p.m. and 6 a.m., the off-peak hours rate is used. If your customer has accumulated fewer than 10 hours of Internet time, your customer is charged $0.05 per minute for IP phone calls. If your customer has accumulated more than 10 hours of Internet time, your customer is charged $0.02 per minute for IP phone calls.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> Resource balance-based rating has limitations for cycle events. If a cycle event is canceled at the end of a quantity discount bracket, BRM bases the cancellation fees on the next quantity discount bracket. For example, if the pricing configuration includes different cycle fees based on the number of elapsed months, you can create a counter for the number of elapsed months: quantity discount bracket 1 for months 0 to 6 (the first six months), and quantity discount bracket 2 for months 6+. If a customer cancels during the sixth month, the cancellation fee is based on quantity discount bracket 2.

Example 1: Rating Based on the Quantity of an Event

To rate events based on their quantity or duration, use either the Continuous (total quantity) or Rate Dependent option. For example, an IP session event uses the following 3 quantity discount brackets to rate the duration of the session:

1. First 10 minutes = $0.12 per minute
2. 10 - 20 minutes = $0.07 per minute
3. Over 20 minutes = $0.05 per minute

Notice that the maximum value in the first two brackets is the same as the minimum value in each subsequent bracket. Always set up balance impact groups this way to avoid errors in your price list.

With continuous rating, the maximum value does not represent an actual quantity; but rather a boundary between ranges that is not actually rated. In this example, the smallest fraction of time less than 10 minutes is rated in the first tier, and the smallest fraction of time greater than 10 minutes is rated by the second tier. Everything up to the maximum value in a range is included in that range, and anything over the maximum value is included in the next range.

Example 2: Rating Based on Total Event Quantities

This example shows how to rate the total number of hours per month for an IP session by using two quantity discount brackets.

- Quantity Discount Bracket 1: 0-10 hours a month = $0.10 a minute
- Quantity Discount Bracket 2: 10 + hours a month = $0.05 a minute

Set up the quantity discount brackets based on the resource balance “ISDN bulk hours". You base charges on the minutes per session and the monthly hourly total. Each bracket requires 2 balance impacts: 1 to track the charge per minute and 1 to track each hour of usage.

> **Note**
>
> Because you track the charges based on the monthly total, you also require an ISDN bulk hours fold event that removes the total hours balance each month.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Example 3: Rating Based on Total Number of Events

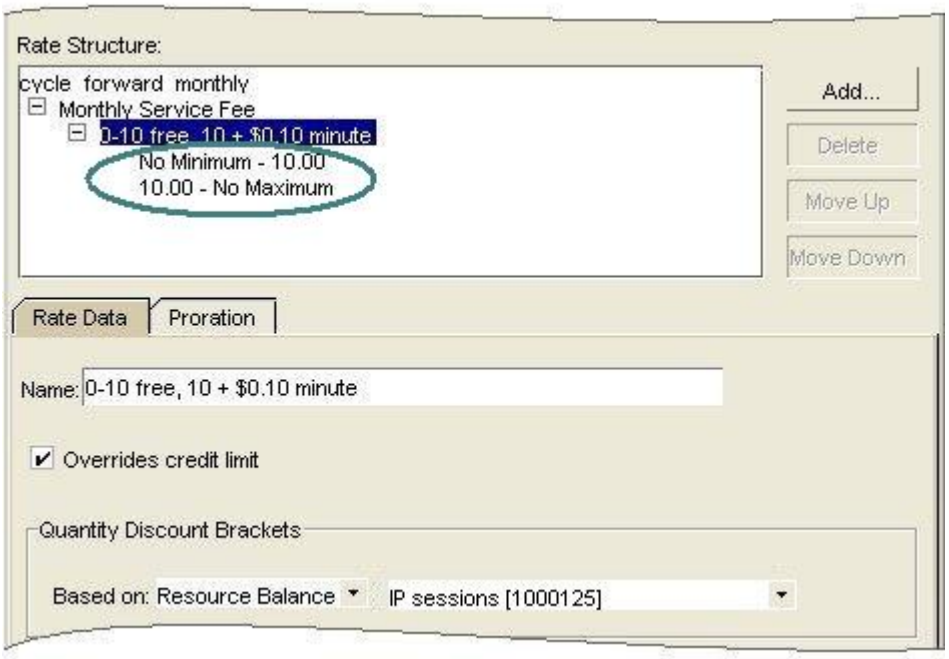
To rate events based on the total number of events that occur over a period of time, use the resource balance for that event. For example, if you charge customers for the total number of IP sessions they have each month, create a custom resource “IP_Sessions" and use it to track the total number of events each month.

For example, an IP Session event is rated based on the number of sessions by using the following 2 quantity discount brackets:

- Quantity Discount Bracket 1: 0-10 sessions = no charge
- Quantity Discount Bracket 2: more than 10 sessions = $0.10 per minute.

Figure 21-3 shows how to set up the brackets in Pricing Center:

Figure 21-3 Rate Structure Based on Number of Events



In the above example, sessions 1-10 are free. Each subsequent session is rated at $0.10 per minute. When the 11th session begins, the resource “IP Sessions" is incremented and the second balance impact containing the scaled amount ($0.10 per minute) is applied to the account balance. This way, when the next session occurs, the second rate is used.

> **Note**
>
> Scaled amounts are applied to the account as soon as the event occurs. Fixed amounts are applied to the account after the event occurs.

The following examples show how to set up the balance impacts for each bracket.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 21-4 shows the first discount bracket.

Set up the first bracket “No Minimum - 10.00" with one fixed amount impact to track the sessions. A scaled dollar impact is not necessary because there is no charge for the first 10 sessions.

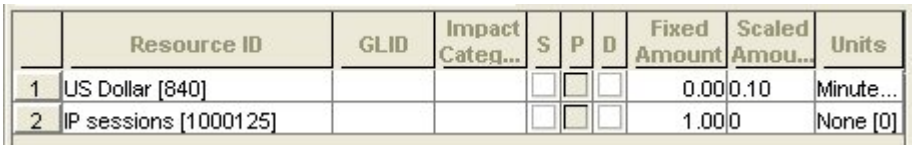
Figure 21-4 Quantity Discount Bracket 1

	Resource ID	GLID	Impact Categ...	S	P	D	Fixed Amount	Scaled Amou...	Units
1	IP sessions [1000125]	undefin...					1.00	0.00	None [0]

Figure 21-5 shows the second discount bracket.

The second bracket “10.00 - No Maximum" requires two balance impacts; one to charge the customer $0.10 per minute and one to track IP sessions.

Figure 21-5 Quantity Discount Bracket 2



	Resource ID	GLID	Impact Categ...	S	P	D	Fixed Amount	Scaled Amou...	Units
1	US Dollar [840]						0.00	0.10	Minute...
2	IP sessions [1000125]						1.00	0	None [0]

Example 4: Rating Based on a Resource Balance Not Affected by the Rated Event

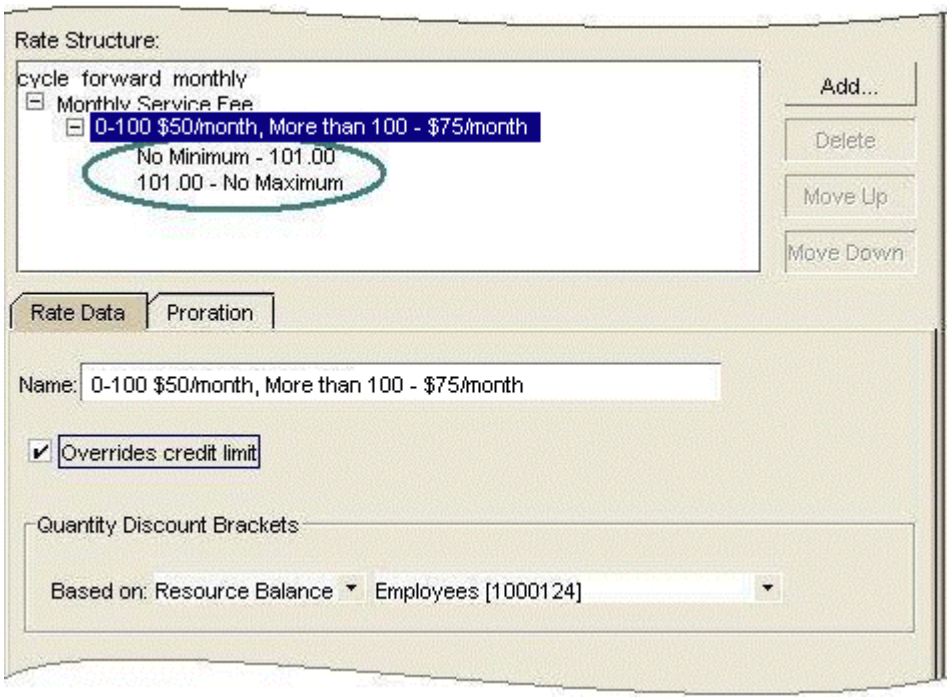
You can charge customers based on resources that do not impact a customer's balance. For example, you can price IP usage based on the total number of employees in a company, or the net worth of a company. These resources are not directly related to the event you are rating. For example, if a company has 100 employees or less, you could charge them $0.03 per IP session; if they have more than 100 employees, you could charge them $0.01 per IP session.

> **Note**
>
> Because the quantity discount brackets are based on a constant resource value, no upper boundary is included in the range. This is different from all other types of rating. In this example, the number of employees is independent of the number of IP sessions.

Figure 21-6 shows how to set up the brackets in Pricing Center:

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 21-6 Rate Structure Based on Number of Employees



Notice that the maximum value for the first bracket is 101 and the minimum value for the second bracket is 101. The maximum value is not included in the valid range and the minimum value is included.

Example 5: Rating an Event As If Using a Single Balance Impact Group

When you use a rate with multiple quantity discount brackets, which bracket is in effect changes as the event quantity changes.

For example, suppose you have the brackets to rate 15 hours of IP usage as shown in Table 21-3:

Table 21-3 Discount Brackets and Rating

Quantity Discount	Event Quantity	Fixed Amount	Scaled Amount	Result of Rating
Bracket 1: 0-5 hours	5	0	$1	$5.00
Bracket 2: 5-12 hours	7	0	$0.75	$5.25
Bracket 3: > 12 hours	3	0	$0.55	$1.65
NA	NA	NA	Total	$11.90

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> Because these fees are assessed on a per-hour basis, each bracket uses only a scaled amount.

Suppose, however, that you want to charge your customer as if all 15 hours were rated by the third group because *the total number of hours falls within that group's range*. In this case, the result is 12 hours @$0.55/hour or a fee of $8.25. To obtain this result, use fixed amounts in the second and third groups to compensate for what was charged in the previous groups as shown in Table 21-4:

Table 21-4 Rating Example

Quantity Discount	Event Quantity	Fixed Amount	Scaled Amount	Result of Rating
Bracket 1: 0-5 hours	5	0	$1	$5.00
Bracket 2: 5-12 hours	7	-$1.25	$0.75	$4.00
Bracket 3: > 12 hours	3	-$2.40	$0.55	-$0.75
NA	NA	NA	Total	$8.25

To determine the correct value for the fixed amount in each group, use the following formula:

$F_c=-Q_p(S_p-S_c)$

Where:

- Fc is the Fixed amount for current bracket.
- Qp is the maximum value of the Quantity range from the preceding bracket.
- Sp is the Scaled amount from the preceding bracket.
- Sc is the Scaled amount from the current bracket.

For example, determine the fixed amounts for the scenario above as follows:

Quantity Discount Bracket 1

A fixed amount is unnecessary.

Quantity Discount Bracket 2

Scaled amount for 5-12 hours = .75

Use the formula above and values from the previous bracket to determine the correct fixed amount:

$F_c=-5(1-.75)]$
$F_c=-1.25$

Quantity Discount Bracket 3

Scaled amount for more than 12 hours = .55

Use the formula above and values from the previous bracket to determine the correct fixed amount:

Copyright © 2025, 2026, Oracle and/or its affiliates.

F_C=-12(.75-.55)]
F_C=-2.4

Copyright © 2025, 2026, Oracle and/or its affiliates.

22

Setting Up Zones for Batch Pipeline Rating

This chapter describes how to set up zones for Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager.

Topics in this document:

- About Zoning
- Setting Up APN Mapping
- Setting Up Prerating
- About Usage Scenario Mapping
- Converting Zone Values for Output
- Setting Up Multi-Segment Zoning

About Zoning

Zoning adds call data to EDRs that can be used for rating. A *zone* is a combination of origin and destination area codes. You create zones to rate calls based on their origin and destination. You can also define zones for different services. For example, you can create zones for rating:

- Local calls.
- National calls.
- International calls.
- Wholesale rating of roaming usage.
- National calls for specific services such as SMS.

There are two types of zones:

- *Standard zones*. These are determined by evaluating the area code of the calling number (the A number) and the called number (the B number). See "About Standard Zones".
- *Geographical zone*s. These are determined by evaluating the distance between the origin and destination of the calls. See "About Geographical Zones".

All zones are mapped to impact categories. You combine impact categories with service usage data to create the final impact categories used by Pipeline Manager as the primary rating attribute. For more information, see "About Impact Categories".

When you configure zones in PDC, you configure the following:

- Standard zones.
- Geographical zones.
- APN maps (for rating GPRS events).
- Zone models, which are collections of zones.
- Impact categories.

Copyright © 2025, 2026, Oracle and/or its affiliates.

About Impact Categories

You use Pipeline Manager impact categories to rate calls based on event attributes such as call origin and destination, and service attributes such as call forwarding. For example, you can create impact categories to rate:

- Local calls.
- Long distance calls.
- Local call forwarding.
- Long distance call forwarding.
- Local mailbox inquiries.
- National mailbox inquiries.

You create an impact category for each zone and usage scenario that you define. A usage scenario describes the type of call, for example, a long-distance mailbox inquiry. You map the following data to impact categories:

- In zone mapping, a zone and service code. The impact category and zone name are the same.
- In usage scenario mapping, a service code, service class, usage class, usage type, and zone.

You must define all possible impact categories before you can use them in a zone model. Before defining impact categories, it is helpful to know the following:

- How many and what kinds of zones you must configure, for example:
  - Domestic zones and international zones.
  - Zones for rating all wireless usage types or for specific usage types only, such as telephony, fax, and SMS.
  - Zones for rating retail usage and roaming usage.
  - No-charge zones such as toll-free and emergency destinations.
- How many and what kind of usage scenarios you must configure, for example:
  - Mailbox inquiries
  - Conference calling
  - Call forwarding
  - Family and friends calls

Your impact category names should reflect the usage scenario. For example, if you offer service-level agreements, you might create three impact categories called **Bronze**, **Silver**, and **Gold**.

For more information on creating usage scenarios, see "Setting Up Usage Scenario Mapping".

About Wholesale and Retail Impact Categories

An impact category can be used for either wholesale rating or retail rating. *Retail rating* applies to usage by your subscribers on your network. *Wholesale rating* applies to usage by visiting subscribers, for example, roaming calls.

Copyright © 2025, 2026, Oracle and/or its affiliates.

You define impact categories according to your retail and wholesale needs. However, when you create impact categories, you do not specify whether they are used for wholesale or retail rating. Instead, you enter the impact category name in the wholesale and retail impact categories or zone fields when configuring zones and usage scenarios. You must enter values for both wholesale and retail impact categories and zones. When calls are rated, the type of impact category used is determined by whether the charge type is wholesale or retail.

About Standard Zones

A *standard zone* is a combination of an origin and destination area code and one or more services such as telephony and SMS. You assign an impact category to each zone that you define. For example, in Table 22-1, telephony origin and destination area codes are mapped to one of three impact categories; **Local**, **Toll**, and **LongDistance**:

Table 22-1 Impact Categories and Destination Area Codes

Origin Area Codes	Destination Area Codes	Destination Area Codes	Destination Area Codes
--	0123	0234	0345
0123	**Local**	**Toll**	**LongDistance**
0234	**Toll**	**Local**	**LongDistance**
0345	**Toll**	**LongDistance**	**Local**
0456	**LongDistance**	**LongDistance**	**Toll**

About Geographical Zones

A *geographical zone* is the distance between the origin and destination of a call. You set up geographical zones to rate calls by distance. This is useful when:

- Customers are located very near the border between two area codes. For example, if a customer in one area code calls a person two blocks away in another area code, you do not want to charge for a long-distance call.
- The distance covered by an area code is very large and you want to use several rates within the same area code. You do this by mapping sets of latitude and longitude coordinates to the same area code in a geographical model.

To set up geographical zones:

- Define geographical zones and map them to distances.
- Create geographical models to:
  - Define the area codes that make up each zone.
  - Map latitude and longitude coordinates to area codes.

Geographical mapping uses the area codes and geographic coordinates to compute the distance and assigns the zone that matches that distance to the call record.

Displaying Zoning Information on an Invoice

You can enter a zone description that can be included in invoices. To do so, in Pricing Center, enter the zone description in the **Description** field in the following dialog boxes:

- Standard Zone

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Geographical Zone
- Usage Scenario Mapping

You can use a maximum of 2,000 characters in the description.

> **Note**
>
> To include zone descriptions on your invoices, you must configure the DAT_Zone and the FCT_USC_Map modules to load invoice description data into memory.

About Pipeline Manager Zone Modules

Zoning is performed by the following function and data modules:

- The FCT_APN_Map module adds Access Point Name (APN) data for the following:
  - To define zones.
  - To adjust impact categories.

  See "Setting Up APN Mapping".
- The FCT_PreRating module calculates zones and creates impact categories. This is the primary zoning module. See "Setting Up Prerating".
- The FCT_USC_Map module adjusts zones based on service attributes. See "Setting Up Usage Scenario Mapping".
- The DAT_Zone module handles zoning data for the zoning function modules.

The following function and iScript modules are used for optional zoning features:

- The FCT_Zone module calculates zones when you run Pipeline Manager for real-time zoning.
- The FCT_SegZoneNoCust maps zone models to segments. Use segment rating for collecting business information, for example, comparing the charges for the same events by using different charges.
- The FCT_MainZoning performs zoning for segment rating. It is used when a pipeline is used only for zoning.
- The ISC_SetSvcCodeRTZoning iScript updates DETAIL.INTERN_SERVICE_CODE EDR field with the customized service code value when different service codes are mapped to the same service type in the pipeline database and a specific service code is used in the zoning table.

Setting Up APN Mapping

Use the FCT_APN_Map module to rate GPRS events. The module runs before zoning or after zoning:

- **Before zoning**. If you rate calls that access a GPRS data service, for example, to download content, there is no B number. In that case, the module runs before zoning. It maps the access point name (APN) to a packet data protocol (PDP) which serves as the B number that can be used for zoning.
- **After zoning**. If you rate GPRS calls made from one phone to another, for example, to exchange calendar appointments, the EDR includes the A number and B number. In this

Copyright © 2025, 2026, Oracle and/or its affiliates.

case, the zoning module has already created an impact category from the A number and B number. The FCT_APN_Map module uses the access point name (APN) to modify the wholesale and retail impact categories.

To configure APN mapping, you do the following:

1. Create APN maps. See "About APN Maps".
2. Configure the FCT_APN_Map module.

About APN Maps

To map APN numbers to PDPs before zoning or to impact categories after zoning, the FCT_APN_Map module reads data from the EDR and evaluates it according to the APN map. An APN map includes one or more mapping rules that specify the data that must be matched to apply the mapping.

You can create multiple sets of APN mapping rules, for example, different mappings for wholesale and retail rating.

You can use APN groups to assign a set of rules to a specific FCT_APN_Map module instance.

You can use the following EDR data to create an APN mapping rule:

- These values are compared with the EDR data when the module runs before or after zoning:
  - APN name (from the Associated GPRS Extension record APN_ADDRESS field)
  - Internal service code
- These values are compared with the EDR data only when the module runs after zoning:
  - Wholesale zone
  - Retail zone

You rank APN mapping rules when you create them. The first rule that matches the EDR data is used. The results of the mapping add the following data to the EDR:

- If the module runs before zoning, it adds the PDP address in the detail record INTERN_B_NUMBER_ZONE field.
- If the module runs after zoning, it adds the following:
  - Impact category
  - Wholesale zone
  - Resale zone

When you run the FCT_APN_Map module after zoning, different EDR fields are used depending on whether you use FCT_PreRate for zoning and rating, or FCT_Zone for zoning only:

- When FCT_PreRate is used (for zoning and rating):
  - The impact category is read from and written to the associated charge breakdown record.
  - The wholesale and retail zones are read from and written to the supplementary zone packet record.
- When FCT_Zone is used (for zoning only), the wholesale and retail zones are read from and written to the detail block of the EDR.

Copyright © 2025, 2026, Oracle and/or its affiliates.

The following tables summarize the data mappings.

Table 22-2 describes the EDR field mapping before zoning when there is no B number.

Table 22-2 Running FCT_APN_Map before Zoning

EDR Field	Compared against This Pipeline Manager Database Field
AssocCharge-APN group	IFW_APN_MAP.APN_GROUP
Detail-Internal Service Code	IFW_APN_MAP.SERVICECODE
Detail-Application	IFW_APN_MAP.ACCESSPOINTNAME

Running FCT_APN_Map after zoning:

For every zone breakdown record the following mapping is performed as shown in Table 22-3:

Table 22-3 Running FCT_APN_Map after Zoning

EDR Field	Compared against This Database Field
AssocZone-APN group	IFW_APN_MAP.APN_GROUP
Detail-Internal Service Code	IFW_APN_MAP.SERVICECODE
Detail-Application	IFW_APN_MAP.ACCESSPOINTNAME
AssocZone-Wholesale Zone Result value	IFW_APN_MAP.ZONE_WS
AssocZone-Retail Zone Result value	IFW_APN_MAP.ZONE_RT

Creating APN Maps

Use PCC to create APN maps and APN groups. APN groups are assigned differently depending on whether you run the FCT_APN_Map module before or after zoning:

- **Before zoning**. Specify the APN map in the **APNGroup** registry setting.
- **After zoning**. Specify the APN map in the zone model.

When you create mapping rules, you use regular expressions.

Setting Up Prerating

Prerating is performed by the FCT_PreRating module. This module uses the data in the EDR to compute zones and assign an impact category to each charge packet.

The module works as follows:

1. It finds the charge to use by reading the charge code in the charge packet.

> **Note**
>
> When an EDR is associated with multiple charges, FCT_PreRating uses the highest priority charge. That is, it searches through all associated charges, from highest priority to lowest priority, until it finds a charge offer that matches the event's criteria. When two charges have matching priority, FCT_PreRating chooses the charge with the earliest start time.

Copyright © 2025, 2026, Oracle and/or its affiliates.

2. In the charge, it finds the zone model.
3. It uses the zone model to calculate the zone and write the impact category to the charge packet. It reads the following information in the EDR:
   - Service code
   - A number
   - B number
   - Call start timestamp

   The data used for determining zones and impact categories is handled by the DAT_Zone module.

To set up prerating, you do the following:

1. Create zone models and impact categories in PDC.

   You can also define zone data for the DAT_Zone module in ASCII files. See "Creating Zone Data Files".

   > **Note**
   >
   > You can use DAT_Zone in file mode in an FCT_Zone/DAT_Zone combination only. Because file-based zoning cannot coexist with pre-rating, DAT_Zone in file mode does not work with FCT_PreRating.

2. Configure the FCT_PreRating module.
3. Configure the DAT_Zone module.

Creating Zone Data Files

Zone data can be defined in ASCII files.

- Each line in a file contains one configuration entry.
- Fields are separated by semicolons (;).

Zone Data File

Zone data files defines the zone model codes in the format:

```
ZoneModelCode;ZoneModelType;GeoModel
```

where:

ZoneModelCode specifies a unique code for the zone model.

ZoneModelType specifies the zone model type; standard (**S**) or geographical (**G**).

GeoModel specifies the geographical link.

Zone data file example:

```
ZM_ADD;S;100000
ZM_GEO;G;100000
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

Standard Zone File

Standard zone file defines standard zone models in the format:

```
ServiceCode;ValidFrom;ValidTo;Origin;OriginDesc;Destination;DestinationDesc;RtZone;WsZone
;ZoneEntryName;ZoneDescription;AltZoneModel
```

where:

ServiceCode is the service code that applies to the zone.

ValidFrom is the date when the standard zone becomes valid, in the format DD.MM.YYYY. For example, **31.12.2004**.

ValidTo is the date when the standard zone becomes inactive. Use the same date format as above.

Origin is the origin area code for the zone.

OriginDesc is the description for the origin.

Destination is the destination area code for the zone.

DestinationDesc is the description for the destination.

RtZone is the impact category for retail rating to apply to the zone.

WsZone is the impact category for wholesale rating to apply to the zone.

ZoneEntryName is the name for the standard zone.

ZoneDescription is the description for the standard zone.

AltZoneModel is the alternate standard zone model that can be used.

> **Note**
>
> The *ZoneEntryName* and *ZoneDescription* fields must be included, even if they are not used.

Standard zone file example:

```
TEL;20030101;;0049;Germany;0049;Germany;NAT_MBI;NAT_MBI;;;
```

Geographical Zone File

Geographical zone file defines geographical zone models in the format:

```
Distance;ServiceCode;ValidFrom;ValidTo;WsZone;RtZone;ZoneEntryName;ZoneDescription;AltZon
eModel
```

where:

Distance is the maximum distance for the geographical zone.

ServiceCode is the service code that applies to the zone.

ValidFrom is the date when the zone becomes valid, in the format DD.MM.YYYY. For example, **31.12.2004**.

Copyright © 2025, 2026, Oracle and/or its affiliates.

ValidTo is the date when the zone becomes inactive. Use the same date format as above.

WsZone is the impact category for wholesale rating to apply to the zone.

RtZone is the impact category for retail rating to apply to the zone.

ZoneEntryName is the name for the geographical zone.

ZoneDescription is the description for the geographical zone.

AltZoneModel is the alternate geographical zone model that can be used.

> **Note**
>
> The *ZoneEntryName* and *ZoneDescription* fields must be included, even if they are not used.

Geographical zone file example:

```
500;*;19990101;;NAT_FIX;NAT_FIX;;;
```

Area Code Coordinate File

Area code coordinate file defines area codes used in a geographical zone in the format:

```
GeoModel;AreaCode;Longitude;Latitude;ValidFrom;ValidTo
```

where:

GeoModel is the description for the geographical link.

AreaCode is the area code to link.

Longitude is the longitude for the area code.

Latitude is the latitude for the area code.

ValidFrom is the date when the geographical link becomes valid, in the format DD.MM.YYYY. For example, **31.12.2004**.

ValidTo is the date when geographical link becomes inactive. Use the same date format as above.

Area code coordinate file example:

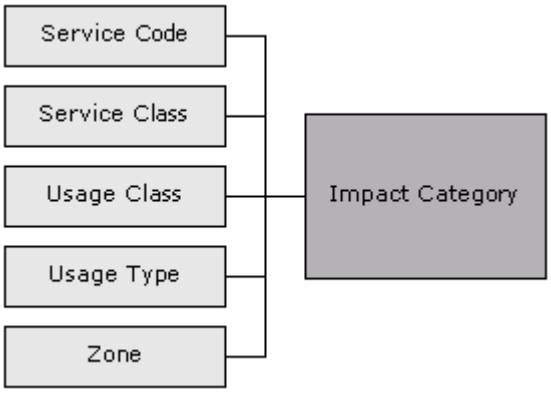
```
100000;049;80;80;19990101;19990115
```

About Usage Scenario Mapping

Usage scenario mapping brings together the data in all EDR fields pertaining to the service code, usage class, usage type, and zone, and then adds the final impact category used for rating the EDR to the charge packet or the detail block, depending on which USC mapping mode you selected as shown in Figure 22-1.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 22-1 Usage Scenario Mapping Factors



The impact category is included as part of the charges.

You define usage scenario mapping to rate differentiated network services, such as mailbox inquiries and friends-and-family discounts.

For example:

- If a customer has a mailbox service, you can create separate impact categories for accessing a mailbox from your network or from a different carrier's network. In this example, the impact category uses two values, the usage class that represents the mailbox, and the zone that the event applies to.
- If you provide a friends-and-family discount, you can create separate impact categories for national friends and international friends.
- If a customer has call forwarding, you can create separate impact categories for calls forwarded to a local number or to a long-distance number.

You can use multiple attributes to define an impact category, or you can use only one attribute.

> **Note**
>
> You can use iScript to define your own attributes to be considered when specifying the final impact category.

Setting Up Usage Scenario Mapping

To set up usage scenario mapping, do the following:

1. Create usage scenario maps. You have two options:
   - You can create usage scenario maps in PCC. In this case, the usage scenario map data is stored in the Pipeline Manager database.
   - You can create a file that defines usage scenario maps. In this case, the FCT_USC_Map module reads the file. For information on creating the file, see "Creating a Usage Scenario Map File".
2. Configure the FCT_USC_Map module.

Copyright © 2025, 2026, Oracle and/or its affiliates.

You specify the mode in which USC mapping is done.

- The **Rating** mode (the default) specifies that USC mapping is done using the zone model from the charge packets. Mapping in this mode provides the impact category for charge packets.
- The **Zoning** mode specifies that USC mapping is done using the zone model from the EDR detail block. Mapping in this mode provides impact categories for the detail block.

3. Configure the DAT_USC_Map module.

   When you use the database interface, you create USC map groups in PCC, which contain the USC maps. You associate USC map groups with zone models. FCT_USC_Map uses the zone model ID in the EDR to identify the associated USC map group. The zone model ID is added to the EDR by the FCT_PreRating module. See "Setting Up Prerating".

> **Note**
>
> If there is no zone value in the EDR, FCT_USC_Map writes an empty string as the value of the zone to the output EDR. This should be considered when you set up the mapping.

> **Note**
>
> You can use semaphore commands with the DAT_USC module to save USC Map data to a file.
>
> - Use **PrintAllUscMapData** to save all of the USC Map data to a file.
> - Use **PrintUscMapDataForZoneModel** to save the data for a specified Zone Model ID.
> - Use **PreCompiledDataDir** to compile USC mapping data and save the data to a specified directory.
> - Use **NumberOfThreads** to specify the number of threads to use when compiling and loading the mapping data.
> - Use **UscGroups** to specify the USC groups for which to load rules.

About Usage Scenario Maps

Usage scenario mapping is performed by the FCT_USC_Map module. To define the final impact category, the module reads data from the EDR and evaluates it according to the usage scenario map. A usage scenario map includes one or more mapping rules that specify the data that must be matched to apply the impact category.

When you create usage scenario maps, you create a mapping rule for each impact category. You can also define the order in which the rules are evaluated. The impact category is derived from the first rule that matches the data. If no matching rule is found in the usage scenario mappings, then no mapping is performed.

When no usage scenario mappings are set up, FCT_USC_Map module uses the default USC map group. To use a default USC map group, you must create it and specify the default USC map group in the FCT_USC_Map registry.

You can use the following data to create a mapping rule:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- The EDR date and time. This data is read from the EDR **StartTimeStamp** field.
- Maximum event quantity, for example, 60 seconds. If the quantity exceeds this value, the map does not apply.
- Minimum and maximum wholesale amount.
- Usage class.
- Usage type.
- Service code.
- Service class.
- Wholesale and retail impact category.

When you create the mapping rules, you use regular expressions.

Creating a Usage Scenario Map File

The USC mapping rules are defined in an ASCII file.

- Each rule consists of a list of fields. The following table describes the meaning of each field.
- Every new line defines an adjustment rule.
- The position of each rule in the file determines its rank. The first matching rule is used.
- Fields are separated by semicolons (**;**).
- Comment lines start with #.
- Empty lines are allowed.

> **Note**
>
> The value of the field rank is ignored. The evaluation order of the rules is given by the order of the rules within the file.

Table 22-4 lists the fields in the file:

Table 22-4 Fields in Usage Scenario Mapping File

Position	Identifier	Description
1	**USCGroup**	Specifies the compare pattern for the source USC group.
2	**Rank**	This parameter is not evaluated. Instead, the rules are evaluated in the order they appear in the file.
3	**ValidFrom**	Specifies the start date for the USC mapping rule. This can either be a date with the format YYYYMMDD or a weekday with an optional timestamp, for example: • 20030524 • SAT • MON16:00 If the field is left empty, the earliest possible date 19010101 is used.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 22-4 (Cont.) Fields in Usage Scenario Mapping File

Position	Identifier	Description
4	**ValidTo**	Specifies the end date for the USC mapping rule. This can either be a date with the format YYYYMMDD or a weekday with an optional timestamp. If the field is left empty, the latest possible date 20370205 is used.
5	**TimeFrom**	Specifies the beginning of the time period (in format HH:MM) the mapping entry is valid on each day of the validity period specified by **ValidFrom** and **ValidTo**. If the field is left empty, 00:00 is used.
6	**TimeTo**	Specifies the end of the time period (format HH:MM) the mapping entry is valid on each day of the validity period specified by **ValidFrom** and **ValidTo**. If the field is left empty, 24:00 is used. Example: To set up a mapping entry that is valid on weekends between 13:00 and 14:00, you must set **ValidFrom**=SAT, **ValidTo**=SUN, **TimeFrom**=13:00 and **TimeTo**=14:00.
7	**Quantity**	Specifies the maximum quantity value for an EDR container. If this maximum is exceeded, the mapping rule will not be used. If this field is left empty or a 0 is specified, the rule is valid for every quantity value. For example, to avoid mapping for calls longer than 60 seconds, set the value to **60**.
8	**MinAocAccount**	Specifies the optional minimal wholesale amount value (the wholesale charge). The map is valid only if the EDR charge value is greater or equal to this setting.
9	**MaxAocAccount**	Specifies the optional maximum wholesale amount value (the wholesale charge). The map is valid only if the EDR charge value is less than or equal to this setting.
10	**SourceUsageClass**	Specifies the compare pattern for the source usage class.
11	**SourceUsageType**	Specifies the compare pattern for the source usage type.
12	**SourceServiceCode**	Specifies the compare pattern for the source service code.
13	**ServiceClass**	Specifies the compare pattern for the source service class.
14	**SourceImpactCategoryWS**	Specifies the compare pattern for the source wholesale impact category
15	**SourceImpactCategoryRT**	Specifies the compare pattern for the source retail impact category.
16	**NewUsageType**	Specifies the resulting usage type for this rule. If this field is left blank or set to **0**, the usage type is not modified.
17	**NewImpactCategoryWS**	Specifies the compare pattern for the wholesale zone which should be mapped.
18	**NewImpactCategoryRT**	Specifies the compare pattern for the retail zone which should be mapped.
19	**Description**	Specifies the rule name. It is not used for rule evaluation.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Converting Zone Values for Output

Use an IRule script to convert the alphanumeric zone values (impact category name) into external result values before the EDR container is written to the output file. Be default, the IMPACT_CATEGORY values from IFW_IMPACT_CAT are written to the output file as the zone value.

Setting Up Multi-Segment Zoning

Use multi-segment zoning to test different zoning models against the same EDR data. Use the FCT_SegZoneNoCust module and the FCT_MainZoning module to perform zoning for multi-segment zoning.

Use the FCT_SegZoneNoCust module to perform zoning when you cannot get customer account data about the EDRs. For example, you might want to use a test system that does not have access to account data. In that case, you configure the FCT_SegZoneNoCust module to find the segment based on the source network ID.

After the FCT_SegZoneNoCust module finds the segment, it adds the zone data for that segment to the EDR. It adds one associated zone breakdown record for each zone in the segment. The FCT_MainZoning module uses this data to perform the zoning and add the results to the zone packets.

To configure multi-segment zoning, configure the FCT_SegZoneNoCust and FCT_MainZoning modules.

Configuring Segments in the FCT_SegZoneNoCust Module

Use the FCT_SegZoneNoCust module **Segments** registry entry to specify the segment to use for each source network. Each rule defines the connection between the source network and the segment. For example:

```
26201 = SegmentD1
26202 = SegmentD2
```

> **Note**
>
> You cannot change these mappings during run time.

Copyright © 2025, 2026, Oracle and/or its affiliates.

23

Real-Time Rating Using Event Attributes

This chapter describes how to define rates in your Oracle Communications Billing and Revenue Management (BRM) price list by using event attributes such as telephone call origins and destinations.

Topics in this document:

- About Using Event Attributes to Rate Events
- Using Event Attributes to Select Rate Plans
- Using Event Attributes to Define Impact Categories
- About Charging for Custom Events and Attributes
- About Real-Time Zoning
- About Setting Up Zones
- Selecting Multiple Rate Plans and Impact Categories
- Customizing Zone Mapping
- Using the Rate Plan Selector in Pricing Center

About Using Event Attributes to Rate Events

If BRM can measure a billable event, it can rate it. It can rate an event based on multiple event attributes, such as the date and time that the event occurred. With attribute-based rating, you can specify additional event attributes to consider for rating, for example, phone call origin and destination.

> **Note**
>
> You can use a rate plan selector to rate cycle events. To do so, you can use account or service attributes, but not event attributes, to select the rate plan.

Using Event Attributes to Select Rate Plans

To use attribute-based rating, you use the Pricing Center *rate plan selector*. The rate plan selector associates a value in an event attribute with a rate plan. For example, by using the event attribute that stores the telephone call destination, you can choose rate plans like this:

- If the call is made to France, use the Calls to Europe rate plan.
- If the call is made to Ghana, use the Calls to Africa rate plan.

Figure 23-1 shows a rate plan selector that chooses a different rate plan depending on call type. In this case, the telephony event includes an attribute named CALL_TYPE, which specifies the type of call, for example, a call made with a calling card. To rate a call made with a calling card differently from a call without a calling card, you assign a different rate plan based on the call type.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 23-1 Rate Plan Selector Configuration for Call Type

Row	EVENT.PIN_FLD_CALL.PIN_FLD_TYPE	+...	Rate Plan	Impact Category
1	5		Calling card	default
2	1		Home phone	default

The event attributes that you use for rating often depend on the source of the usage data. For example, when rating telephone calls, some PBX-to-gateway/gatekeeper systems do not supply the ANI value. In that case, you can use the CALL.ORIG value instead.

Using Event Attributes to Define Impact Categories

You use impact categories to apply different balance impacts for the same rate plan. For example, you can assign a set of telephone calls and destinations to a single rate plan:

Rate plan: IP Telephony

Impact category: USA to Europe

Balance impact: .10

Impact category: USA to Asia

Balance impact: .15

Notice that the same rate plan is used for all calls, but the impact category specifies different balance impacts.

Even though you are using a single rate plan here, you must use the rate plan selector to assign impact categories. Therefore, you must choose **Rate plan selector** when creating the product, not **Single rate plan**.

You use the rate plan selector to associate event attributes with impact categories.

The event attributes ANI (call origin) and DNIS (call destination) are entered as country codes:

- 1 = USA
- 33 = France
- 49 = Germany
- 81 = Japan
- 86 = China

A call from the USA to France might have this call origin and destination:

- Origin: 15555121212
- Destination: 3372621234

BRM matches the numbers in the rate plan selector with the telephone numbers. Since only country codes are used, all calls from each country match the numbers in the rate plan selector. For example, “3372621234" matches “33."

In the balance impacts for this rate plan, the impact categories define how much to charge.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> In **Balance Impacts**, you can use asterisk (*) as a wildcard character in the **Impact Category** column to apply the balance impact to all impact categories defined for the rate plan.

Using the above examples, here is how BRM assigns a balance impact to a call:

1. The customer makes a call from the USA to France.
2. During rating, BRM finds the call origin and destination data in the event.
3. BRM looks through the data in the rate plan selector to find entries that have a call origin of USA and a call destination of France. This information tells BRM which rate plan to use, and which impact category to use.
4. BRM looks in the IP Telephony rate plan for the balance impact.
5. BRM looks through the IP Telephony balance impacts to find one that uses the USA_Europe impact category. BRM uses that balance impact to calculate the charge for the call.

About the Default Impact Category

You use the default impact category to apply a balance impact when none of the other impact categories can be used. The default impact category provides a fail-safe; without it, some events might not be rated at all.

The default impact category name is **default**.

Setting the impact category to **default** applies the balance impacts of the rate plan according to standard rating attributes, and ignores all impact category conditions.

Creating Impact Categories

To make impact categories available to Pricing Center, you edit the **pin_impact_category** file, then run the **load_pin_impact_category** utility to load the contents of the file into the **/config/impact_category** object in the BRM database.

> **Note**
>
> The **load_pin_impact_category** utility requires a configuration (**pin.conf**) files.

1. Edit the **pin_impact_category** file in *BRM_home***/sys/data/config**. The **pin_impact_category** file includes examples and instructions.

> **Note**
>
> The **load_pin_impact_category** utility overwrites existing impact categories. If you are updating impact categories, you cannot load new impact categories only. You must load a complete set of impact categories each time you run the **load_pin_impact_category** utility.

Copyright © 2025, 2026, Oracle and/or its affiliates.

2. Save the **pin_impact_category** file.
3. Use the following command to load the file into the database:

   ```
   load_pin_impact_category pin_impact_category
   ```

   If you are not in the same directory as the **pin_impact_category** file, include the complete path to the file, for example:

   ```
   load_pin_impact_category BRM_home/sys/data/config/pin_impact_category
   ```

   For more information, see "load_pin_impact_category".
4. Stop and restart the Connection Manager (CM). If necessary, stop and restart Pricing Center.

To verify that the impact categories were loaded, you can display the **/config/impact_category** object by using the Object Browser, or use the **robj** command with the **testnap** utility.

Creating Unassigned Impact Categories

You can include more impact categories in the **pin_impact_category** file than you need. You do not have to assign a balance impact to all impact categories.

For example, you can plan ahead for origin/destination pairs that you will add later and create the impact categories that they will use.

About Charging for Custom Events and Attributes

BRM supports a default set of event attributes, such as geographical zones, quality of service, and so on. You can define charges for these attributes by using the Rate Plan Selector.

If you want to rate custom attributes of events or non-event attributes or if your rating guidelines are more complex than the defaults provided with BRM, you must modify the PCM_OP_ACT_POL_SPEC_RATES policy opcode.

For example, you can rate administrative events or non-event attributes such as an account attribute or a profile attribute. You can modify the policy to rate percentage values or values containing a greater than or less than operator. You can rate an event one way if an attribute is greater than 10, and another way if it is less than 10.

Charging for Custom Events and Attributes

To charge for custom events and attributes:

1. Modify the PCM_OP_ACT_POL_SPEC_RATES policy opcode or write a custom policy opcode to analyze the event data and assign the rate plan and impact category.

   See the policy source file of the PCM_OP_ACT_POL_SPEC_RATES policy opcode for a sample implementation.

   For information on how generic rating works, see the description of PCM_OP_ACT_USAGE.
2. Edit the **pin_spec_rates** file in *BRM_home***/sys/data/config** to associate the opcode to the event type that it rates.

   The **pin_spec_rates** file includes examples and instructions.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> The **load_pin_spec_rates** utility overwrites the existing setup for administrative events charges. If you are updating a set of administrative events charges, you cannot load new charges only. You load complete sets of charges each time you run **load_pin_spec_rates**.

3. Create a configuration file for the "load_pin_spec_rates" utility.
4. Run the **load_pin_spec_rates** utility to load the contents of the **pin_spec_rates** file into the database.

   ```
   load_pin_spec_rates pin_spec_rates_file
   ```

5. Edit the **pin_impact_category** file in the *BRM_home*/**sys/data/config** directory to define impact categories for the event.

   The **pin_impact_category** file includes examples and instructions.

> **Note**
>
> When you run the **load_pin_impact_category**, it overwrites the existing impact categories. If you are updating a set of impact categories, you cannot load new impact categories only. You load complete sets of impact categories each time you run the **load_pin_impact_category** utility.

6. Run the "load_pin_impact_category" utility to load the impact categories into the database.

   ```
   load_pin_impact_category pin_impact_category_file
   ```

7. If you use pipeline batch rating, add the custom cycle event to the following locations:
   - The EAI payload configuration file.
   - Your system's event notification list. The information you must add to the list is in the **pin_notify_ifw_sync** file.
8. In Pricing Center, create the rate plans that can be used to rate an event that uses custom event analysis.
9. In your pricing plan, assign a rate for the impact category.
10. Use the Pricing Center to create products that use the rates defined in your custom opcodes. Ensure that the products use custom event analysis.

Using Custom Fields with Real-Time Rating

If you add rating-related custom fields to BRM, you must make these fields available to the rating opcodes. For instructions, see"Making Custom Fields Available to Your Applications" in *BRM Developer's Guide*.

About Real-Time Zoning

You use zones to group event attribute values into manageable categories. For example, to apply the same rate plan or impact category to all area codes in Northern California, you can create a zone that includes all Northern California area codes. Another zone might include all Southern California area codes. You then might include both zones in a zone for all California area codes, as shown in the following example:

Copyright © 2025, 2026, Oracle and/or its affiliates.

California zone

Northern California

408

415

650

Southern California

213

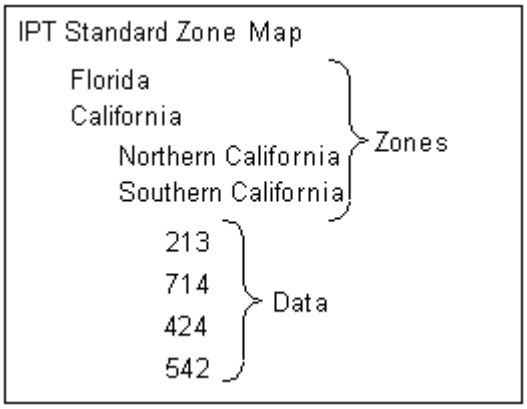
714

424

A zone map contains a hierarchy of zones and *data*, such as phone numbers, IP addresses, and Quality of Service (QoS) data.

Figure 23-2 shows a zone map with multiple nested zones. The data consists of area codes:

Figure 23-2 Zone Map with Nested Zones



If you use a zone map for an event attribute, all of that type of event attribute must be defined with a zone or data from that zone map. You cannot mix hard-coded values in a rate plan selector that uses zones.

For more information on zone, see "About Zoning".

About Setting Up Zones

When you use zone maps to rate events, BRM looks for the product to use for rating an event. If the product is configured to use a pipeline zone model, the real-time rating opcodes send the event to a real-time zoning pipeline to retrieve the zoning information. If the product is not configured to use a pipeline zone model, the real-time rating opcodes search the BRM database for the zoning information for the event.

You can define zones to rate events by using Pricing Center or Zone Mapper:

- To use zone information for rating both real-time and batch rating, use Pricing Center. Zone maps defined using Pricing Center are stored in the Pipeline Manager database.

Copyright © 2025, 2026, Oracle and/or its affiliates.

If you use both real-time and batch processes for rating events, or if you are a new BRM user, define your zones by using Pricing Center.

See "About Zoning" and "Setting Up Zones by Using Pricing Center".

- To use zone information for rating only real-time events, use Zone Mapper in Pricing Center. Zone maps defined by using Zone Mapper are stored in the BRM database.

  If you have already defined zone maps in BRM by using Zone Mapper, you can continue to use your existing zone maps.

  See "About Setting Up Zones by Using Zone Mapper ".

Pricing Center displays the zone-map names and impact categories from both the Pipeline Manager database and the BRM database. You can use the zone maps stored in either database to configure price plans.

Real-Time Zoning Pipeline Architecture

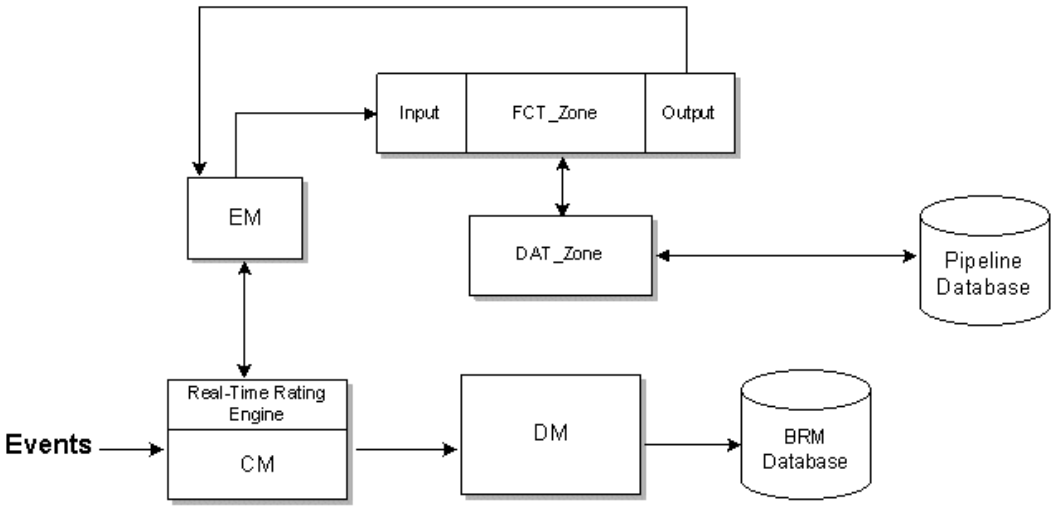
A real-time zoning pipeline processes events as follows:

1. When the CM receives an event with a zoning request, the real-time rating opcodes check the event and the product that rates the event.
2. If the product is configured for pipeline zoning, the real-time rating opcodes extracts the calling number, called number, service calls, event time, and zone map name from the event and product flist.
3. BRM sends the event flist in a request to the NET_EM module to get the impact category from the Pipeline Manager database for the event. The CM sends the request for specific zone map data, such as standard zone-map data and geographic zone-map data. See "Configuring the NET_EM Module for Real-Time Processing" in *BRM Pipeline Manager Administration Guide*.
4. When an event flist is passed to a real-time zoning pipeline, the input module in the pipeline converts the event data from the flist format used by real-time rating opcodes to the EDR format used in the pipeline.
5. The FCT_Zone module processes the zoning EDR and uses the DAT_Zone module to search for the zone map for the event.
6. When zoning is complete, the EDR, now containing zoning information for the event, is converted back into flist format by the output module in the pipeline. The output module also passes the event back to the real-time rating opcodes.
7. If a correct zone map for the event is not found in the Pipeline Manager database, the pipeline zoning module returns an empty flist to the CM. The CM logs the error and the event is not rated.

Figure 23-3 illustrates the data flow for real-time pipeline zoning:

Copyright © 2025, 2026, Oracle and/or its affiliates.

Figure 23-3 Data Flow for Real-Time Pipeline Zoning



Setting Up Zones by Using Pricing Center

1. In Pricing Center, set up your zone models, which are stored in the Pipeline Manager database.

   > **Note**
   >
   > Ensure that the zone map names are unique between both the BRM and Pipeline Manager databases.

2. Configure the NET_EM to receive events from the real-time rating opcodes.
3. Configure the CM to point to the NET_EM.
4. Configure the INP_Realtime module.

   Use this entry for the OpcodeMapping entry:

   ```
   OpcodeMapping = ./formatDesc/Formats/Realtime/zonemap_event.xml
   ```

5. Configure the OUT_Realtime module.
6. Configure the FCT_Zone module to specify a connection to the DAT_Zone module.
7. Configure the DAT_Zone module to process real-time events.

About Setting Up Zones by Using Zone Mapper

You use Zone Mapper to create zones and zone maps, which are stored in the BRM database. Zone Mapper is a component of Pricing Center. You can create multiple zone maps.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> You can use a rate plan selector to rate cycle events. To do so, you can use account or service attributes, but not event attributes, to select the rate plan.

> **Note**
>
> A zone map name can include a maximum of 255 characters.

About Defining Zone Maps for Products

Each product can have only one zone map. However, one zone map can be used by multiple products. You can create multiple zone maps for one system or for branded systems.

> **Note**
>
> All brands in a database can use any of the zone maps stored in that database.

Options for Matching Event Attribute Values

When you create zones using Zone Mapper, you specify how BRM matches a value in an event with the data in the zone map. You have two choices:

- **Prefix:** BRM searches for the value in the zone map that best matches the event. For example, in an IP telephony service, if the customer calls 202-555-5678, and the zone map includes 202, BRM will find the zone with 202. If the zone map includes both 202 and 202-555, BRM will find the zone that includes 202-555.
- **Exact Match:** BRM searches only for the value in the zone map that exactly matches the event. For example, if the customer calls 202-555-5678, BRM looks only for a zone map that includes 202-555-5678.

Changing and Deleting Zone Maps

You can either change a zone name, modify the data for a zone, or delete a zone map entirely. When you delete a zone map that you used to define a product in Pricing Center, you must also remove the map from that product. When you delete a zone map, you also delete the zones that belong to it.

> **Note**
>
> - Deleting a zone map permanently removes the map from the BRM database.
> - If customers have already purchased a product that includes a zone map that you are deleting, you must cancel the product that includes the old zone map, and purchase the product that includes the new zone map.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Selecting Multiple Rate Plans and Impact Categories

You can use the rate plan selector to choose rate plans and assign impact categories at the same time.

Figure 23-4 shows a rate plan selector that selects rate plans based on three event attributes:

- Call origin
- Call destination
- Call type

Figure 23-4 Rate Plan Selector

Row	D_CA...	D_CAL...	EVENT.PIN_FLD_CALL.PIN_FLD_TYPE	+...	Rate Plan	Impact Category
1	US	Europe;France	5		Calling card	US_to_Europe
2	US	Europe;Germany	5		Calling card	US_to_Europe
3	US	Europe;France	1		Telephony	US_to_Europe
4	US	Europe;Germany	*		Telephony	US_to_Asia

Customizing Zone Mapping

The zone mapping opcodes perform zoning for real-time rating.

How Zone Mapping Works

To manage zones, for example, create, delete, and update zone maps, use PCM_OP_ZONEMAP_COMMIT_ZONEMAP. Zone maps are stored in **/zonemap** objects.

By default, PCM_OP_ZONEMAP_COMMIT_ZONEMAP takes an array of zone maps as its input, each of which represents the new, modified, or deleted zone map. In the case of a deletion, the opcode marks the zone map to be deleted in the input flist.

The PCM_OP_ZONEMAP_COMMIT_ZONEMAP output is an flist that links zone maps and products.

To retrieve zone map information from the database, use PCM_OP_ZONEMAP_GET_ZONEMAP. The default implementation enables you to retrieve zone maps in several ways:

- You can use it get all the zone maps by providing a zone map type POID
- You can use it get a specific zone map by providing the zone map name
- You can get the zone map associated with a product by providing the valid product POID

PCM_OP_ZONEMAP_GET_ZONEMAP is called by the Zone Mapper to retrieve zone map data and displays it in the user interface. It is also called by the zone map search opcode, PCM_OP_ZONEMAP_POL_GET_LINEAGE.

Customizing Zone Mapping

Use the following opcodes to customize zone mapping:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- PCM_OP_ZONEMAP_POL_GET_LINEAGE. See "Finding Zone Maps".
- PCM_OP_ZONEMAP_POL_GET_ZONEMAP. See "Getting Zone Maps from the BRM Database".
- PCM_OP_ZONEMAP_POL_SET_ZONEMAP. See "Saving Zone Map Data".
- PCM_OP_RATE_POL_EVENT_ZONEMAP. See "Getting Zone Maps and Impact Categories from the Pipeline Manager Database".

Finding Zone Maps

To search in a zone map for data associated with a given string, use PCM_OP_ZONEMAP_POL_GET_LINEAGE.

You supply a string and a zone map name. It then searches the zone map for the given string and returns the matching node with all ancestors of the matching node (the *lineage*).

This opcode calls the zone-based rating load opcode, PCM_OP_ZONEMAP_GET_ZONEMAP.

Getting Zone Maps from the BRM Database

To customize which zone map data to retrieve from the database, use PCM_OP_ZONEMAP_POL_GET_ZONEMAP. This opcode provides zone maps to display in the Zone Mapper.

This opcode retrieves the original zone map data from the BRM database, and depending on the calling program, it converts the binary format of zone map data into a readable flist, or it returns the binary format to be used in the lineage search.

The input flist must include either the PIN_FLD_POID for routing purposes or PIN_FLD_NAME to search by name.

Saving Zone Map Data

To customize how to save zone map data, use PCM_OP_ZONEMAP_POL_SET_ZONEMAP. This opcode saves zone map in the BRM database when you commit zone maps in the Zone Mapper.

This opcode retrieves the original zone map data from the BRM database, and depending on the calling program, it converts the binary format of zone map data into a readable flist, or it returns the binary format to be used in the lineage search.

The input flist must include either the PIN_FLD_POID for gateway routing purposes or PIN_FLD_NAME to search by name.

Getting Zone Maps and Impact Categories from the Pipeline Manager Database

To get zoning data from the Pipeline Manager database, use the PCM_OP_RATE_POL_EVENT_ZONEMAP policy opcode. This opcode returns the zone map name and impact category for an event from the Pipeline Manager database.

You can customize this policy to add new event classes that you have created, if those event classes use a real-time zoning pipeline.

PCM_OP_RATE_POL_EVENT_ZONEMAP performs the following functions:

1. Reads the event storable class.

Copyright © 2025, 2026, Oracle and/or its affiliates.

2. Based on the service, it retrieves the calling number, the called number, and the zone map name from the event.
3. Prepares the input flist for a real-time zoning pipeline input module.
4. If it finds a matching impact category for the event zone, returns the impact category for the event.
5. If it doesn't find a matching impact category, returns an empty string and logs an error.

PCM_OP_RATE_POL_EVENT_ZONEMAP returns the zone map name and impact category for the event. If the opcode fails, it returns an empty string and logs an error.

Using the Rate Plan Selector in Pricing Center

Before you use the rate plan selector, you must create impact categories. See "Creating Impact Categories".

When you use a rate plan selector in Pricing Center, you start with an empty rate plan selector. After you provide a name for the rate plan selector, and (optionally) choose a zone map, you specify the attribute type and attributes you want to use.

To choose the attribute type, you click the appropriate option. To choose attributes, you select from a list of all attributes in the object.

To add rate plans and impact categories, you choose from lists. You can create a rate plan if it is not on the list, but impact categories must already be defined.

Set the impact category to **default** to apply the balance impacts of the rate plan according to standard rating attributes, and ignore all impact category conditions.

Using Wildcards to Match Event Data

In a rate plan selector, use an asterisk (*) as a wildcard as follows:

- In attribute columns, enter an asterisk to match any value in an event.
- In the **Impact Category** column, do not use an asterisk. BRM does not support its use as a wildcard in this column.

  See "Using Event Attributes to Define Impact Categories" for information about using wildcard characters in rate plan balance impacts.

Specifying the Priority for Attributes

The order of the rows in the rate plan selector determines the order in which rate plans are processed. The top row has the top priority. For example, you can specify to use one rate plan for one value of an attribute, and another rate plan for all other values. To do this, you give the value the top priority as shown in Figure 23-5:

Figure 23-5 Rate Plan Selector

Row	EVENT.PIN_FLD_CALL.PIN_FLD_TYPE	+...	Rate Plan	Impact Category
1	5		Calling card	default
2	*		Non-card	default

When you use multiple attributes, they are processed in the order from left to right.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Maintaining the Order of Rate Plan Selectors

When you create a rate plan selector in Pricing Center or an external client application, you can specify the order in which rate plans are processed. However, this order is not preserved when the rate plan selector is committed to the database. By default, the PCM_OP_PRICE_SET_PRICE_LIST opcode reorders the rate plan selectors to increase performance and reduce buffer size.

You can configure BRM to maintain the order passed in by Pricing Center or the PCM_OP_PRICE_SET_PRICE_LIST opcode by modifying the Connection Manager (CM) configuration file.

To do so, you:

1. Open the CM configuration file (*BRM_home***/sys/cm/pin.conf**).
2. Add the following entry:

   ```
   - fm_price rps_no_reorder 1
   ```

   where:

   - **1**: Leaves rate plan selectors in the order in which they appear in Pricing Center or the PCM_OP_PRICE_SET_PRICE_LIST opcode.
   - **0**: Specifies to reorder rate plan selectors before committing them to the database. This increases performance and reduces the buffer size. This is the default.
3. Save the file.
4. Stop and restart the CM.

Copyright © 2025, 2026, Oracle and/or its affiliates.

24

Migrating Pricing Data from Legacy Databases

This chapter describes the process of migrating legacy pricing and discounting data to the Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager database.

Topics in this document:

- About Migrating Legacy Data
- Guidelines for Mapping Legacy Data
- Mapping Legacy Data
- About the Types of Objects to Migrate
- Working with and Modifying the XSD
- Before Loading Legacy Data
- Configuring the Registry File for LoadIfwConfig Utility
- Loading Legacy Data into the Pipeline Manager Database with LoadIfwConfig
- Deleting Data from the Pipeline Manager Database with LoadIfwConfig
- Updating Data with the LoadIfwConfig Utility
- Exporting Data from the Database with the LoadIfwConfig Utility
- Troubleshooting

To migrate legacy data, you should be familiar with database administration tasks and must:

- Have in-depth knowledge of your legacy database.
- Have in-depth knowledge about the Pipeline Manager database.
- Know XML programming.
- Have a good understanding of XSD. See "Working with and Modifying the XSD".
- Understand the objects that are supported by the Pipeline Manager database and know the order in which the objects must be loaded into the Pipeline Manager database. For information, see "About the Types of Objects to Migrate".

> **Note**
>
> Pipeline Configuration Manager is an optional component.

About Migrating Legacy Data

The Pipeline Configuration Manager is a set of scripts and utilities you can use to load data into the Pipeline Manager database.

Use the **LoadIfwConfig** utility to load pricing data into the Pipeline Manager database. You can migrate legacy data by adding the data to an XML file and running the **LoadIfwConfig** utility to load the data.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Any type of legacy database such as Oracle can be migrated to the Pipeline Manager database by using the **LoadIfwConfig** utility.

You use the **LoadIfwConfig** utility to migrate each table of your legacy database to the Pipeline Manager database. For more information, see "Loading Legacy Data into the Pipeline Manager Database with LoadIfwConfig".

During data migration, if the legacy system and Pipeline Manager are run in parallel, keep the databases synchronized by migrating data every time you make a change in the legacy database.

Overview of the Migration Process

The migration process follows these steps:

1. Planning the migration.

   This step includes comparing the data in the legacy database and the Pipeline Manager database and figuring out how to migrate the data from one database structure to another. See "Guidelines for Mapping Legacy Data".

2. Extracting legacy data to an XML file by using a data transformation program. See "About Migrating Legacy Data ".

3. Loading data into the Pipeline Manager database by using the **LoadIfwConfig** utility. See "Loading Legacy Data into the Pipeline Manager Database with LoadIfwConfig".

Legacy Data XML File Example

Legacy data representation in the XML file

```
<?xml version="1.0" encoding="UTF-8"?>
<IFW xmlns ="http://www.portal.com/tools/pricing" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="http://www.portal.com/tools/pricing
IfwConfig.xsd">
<UOM UnitsOfMeasurement="hp1" Name="abc"/></IFW>
```

Guidelines for Mapping Legacy Data

You map the legacy pricing data to the pipeline pricing data in an XML file. This XML structure is defined in the XSD (**IfwConfig.xsd**). See "Working with and Modifying the XSD".

The structure of legacy pricing data is almost always different from the structure used by Pipeline Manager. For example, the legacy data might have pricing element equivalent to a pipeline charge, but not the equivalent of a pipeline charge version.

When you map legacy data, compare the structure of your legacy data against the structure of the Pipeline Manager data. Table 24-1 shows a sample comparison:

Table 24-1 Comparing Legacy and Pipeline Manager Data

Legacy Data	Pipeline Manager Data
Charge	Charge
No equivalent	Charge version
Charge configuration	Charge configuration
Charge model	*No equivalent*

Copyright © 2025, 2026, Oracle and/or its affiliates.

In this example:

- You must create charge versions. This data might be in another part of the legacy data; for example, the version information might be included in the legacy charge itself.
- You must map the data in the legacy charge model to one of the pipeline charge elements.
- The legacy data and the Pipeline Manager data have charges and charge configuration. However, you must ensure that those elements include the same data.
- The XML file you create must use the structure defined in the XSD.

If you cannot directly map the data, then export the legacy data to flat files, and run a data transformation program on the flat files to produce data that can be loaded into the Pipeline Manager database. You can use any third-party tool for this purpose, depending on the type of legacy database. XML is a widely accepted programming language so many parsers and tools are available to transform data to XML.

Charge XML File Example

The following examples show the charge structure from the XSD file, and a charge in the XML file:

Charge definition in the XSD file

```
<xs:element name="RatePlan">
  <xs:annotation>
    <xs:documentation>...</xs:documentation>
  <xs:annotation>
  <xs:complexType>
    <xs:attribute name="RatePlanCode" type="xs:string" use="required"/>
    <xs:attribute name="Name" type="xs:string"/>
    <xs:attribute name="Status" default="D">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:length value="1"/>
          <xs:minLength value="1"/>
          <xs:maxLength value="1"/>
          <xs:enumeration value="S"/>
          <xs:enumeration value="T"/>
          <xs:enumeration value="A"/>
          <xs:enumeration value="D"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="CurrencyCode" type="xs:string" use="required" fixed="USD"/>
    <xs:attribute name="IsSplitting" type="xs:boolean" default="0"/>
    <xs:attribute name="Type" default="R">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:length value="1"/>
          <xs:minLength value="1"/>
          <xs:maxLength value="1"/>
          <xs:enumeration value="W"/>
          <xs:enumeration value="R"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="TaxTreatment" type="xs:boolean" use="required"/>
    <xs:attribute name="CalendarCode" type="xs:string" use="required"/>
    <xs:attribute name="UTCTimeOffset" type="xs:string" default="+0100"/>
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
  </xs:complexType>
</xs:element>
```

Charge representation in the XML file

```
<RatePlan
RatePlanCode="ratepc"
Name="rateplanc"
Status="D"
CurrencyCode="USD"
IsSplitting="0"
Type="R"
TaxTreatment="1"
CalendarCode="cal2001"
UTCTimeOffset="+0100"/>
```

Mapping Legacy Data

You extract the legacy data to a file by using a data conversion program. After the data is extracted from the legacy database, you migrate the data into an XML file. (You can also extract the data directly into an XML file.) You can insert different types of data, objects, and attributes in a single XML file.

To map legacy data, create an XML file that uses the structure defined in the XSD. Then migrate the objects in the specified order. See "About the Types of Objects to Migrate".

About the Types of Objects to Migrate

The objects that can be migrated to the Pipeline Manager database are defined in the XSD.

Table 24-2 shows the objects that can be migrated to the Pipeline Manager database and the prerequisite objects required for some objects. The prerequisite objects are the parent objects, which have objects dependent on them.

You must load child objects in the order shown in the table, as data in some objects depends on the existence of other objects. You load parent objects before child objects. However, sometimes a parent object can be loaded after loading a child object if the field in the child object that relates to parent object is not a mandatory field. For example, the Geomodel object is a parent of the Zonemodel object. However, the **Geomodel** field in the Zonemodel object is not a mandatory field, so you can load the Zonemodel object first.

Table 24-2 Order to Be Used in Loading Child Objects

Objects	Parent Object Prerequisites
Edrcdesc	None
Edrcfield	Edrcdesc
Aliasmap	Edrcfield Edrcdesc
Calendar	None
Special day	Calendar
Pipeline	Edrcdesc
Uom	None
Currency	None

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 24-2 (Cont.) Order to Be Used in Loading Child Objects

Objects	Parent Object Prerequisites
Resource	Currency
Taxcode	None
Taxgroup	None
Tax	Taxgroup Taxcode
GLaccount	Taxcode
Revenuegroup	None
Rum	None
Rumgroup	None
Rumgrouplnk	Rumgroup Rum Uom
Service	GLaccount Revenuegroup Rumgroup
Serviceclass	Service
Refmap	None
Mapgroup	None
Servicemap	Mapgroup Service
Usageclass	None
Usageclassmap	Usageclass Mapgroup
Uscgroup	None
Rscgroup	None
Apngroup	None
Zonemodel	Geomodel Apngroup
Impactcat	None
Geomodel	Ruleset
Geoarealnk	Geomodel
Geozone	Zonemodel Service Impactcat
Standardzone	Service Zonemodel Impactcat
Usagetype	None

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 24-2 (Cont.) Order to Be Used in Loading Child Objects

Objects	Parent Object Prerequisites
Uscmap	Usagetype Zonemodel Uscgroup Impactcat
Apnmap	Apngroup Impactcat
Discarding	Pipeline
Daycode	None
Timeinterval	None
Timezone	None
Timemodel	None
Timemodellnk	Timezone Daycode Timeinterval Timemodel
Pricemodel	None
Pricemdlstep	Pricemodel Resource Rum GLaccount Revenuegroup
Discountmodel	None
Dscmdlver	Discountmodel
Discountmaster	None
Discountdetail	Discountmaster
Dsctrigger	None
Dsccondition	Dsctrigger
Discountrule	Discountmaster
Discountstep	Discountrule
Dscmdlcnf	Discountrule Dsctrigger Dscmdlver
Rateplan	Calendar Currency
Rateplanver	Zonemodel Rateplan

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 24-2 (Cont.) Order to Be Used in Loading Child Objects

Objects	Parent Object Prerequisites
Rateplancnf	Service Rateplanver Pricemodel Timezone Timemodel Impactcat
Rateadjust	Rateplanver
Specialdayrate	None
Specialdaylnk	Rateplanver Specialdayrate
Scenario	Edrcdesc
Uommap	Uom Rum
Exchangerate	Currency
Rscmap	Serviceclass Rscgroup
Rule	None
Ruleitem	None
Ruleset	None
Rulesetlist	Ruleset
Sla	Uscgroup Rscgroup Ruleset
Splittingtype	Pipeline
IcDaily	None
IcDailyalternate	None
Condition	Scenario Edrcfield
Icproduct	None
Networkoper	Taxgroup Currency
Networkmodel	Rateplan Networkoper Currency
Icproductgrp	Networkmodel
Icproductcnf	Icproduct Icproductgrp

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 24-2 (Cont.) Order to Be Used in Loading Child Objects

Objects	Parent Object Prerequisites
Icproductrate	Icproduct Networkoper Rateplan Networkmodel
Noproduct	Networkoper Currency
Noproductcnf	ICproduct Timezone Noproduct Impactcat
Nosp	Mapgroup
Poi	None
Segment	None
Segratelnk	Rateplan Segment
Segzonelnk	Segment Zonemodel
Ciberocc	Networkoper
Socialnumber	None
Seqlogout	None
Dbversion	None
Iscript	None
Tam	None
Seqcheck	None
SeqlogIn	None
Duplicatecheck	None
Csstate	None
Destindesc	None
Classitem	None
Class	None
Queue	None
Changeset	None
Switch	Networkoper
Trunk	Switch Networkoper
Grouping	Scenario Edrcfield
Groupingcnf	Grouping Class

Copyright © 2025, 2026, Oracle and/or its affiliates.

Table 24-2 (Cont.) Order to Be Used in Loading Child Objects

Objects	Parent Object Prerequisites
Classcon	GroupingCnf
Poiarealnk	Networkmodel Poi
Nobillrun	Networkoper Networkmodel
Aggregation	Scenario Edrcfield
Classlnk	Class Classitem
Classconlnk	Classitem Classcon
Dictionary	Queue
Trunkcnf	Trunk IcProductgrp Networkmodel Poi Switch Networkoper
Lergdata	Geoarealnk
Dscbalimpact	Discountstep
Cslock	Changeset
Csaudit	Changeset
Csreference	Changeset

Working with and Modifying the XSD

The XSD describes the structure of the XML document. The XML file you create must comply with the structure defined in the XSD.

The XSD defines the following items for an XML file:

- The elements and attributes, their data types, and the default and fixed values for the elements and attributes.
- Elements that are child elements, and the number and order of the child elements.
- If an element can be empty or can include text.

You can also modify the XSD to provide default values for certain attributes such as status, and provide domain constraints to the attributes. For example, the status can be A=active, D=de-active, T=test, or S=simulation. If the status S is not valid, you can restrict the XSD to allow only the status A, D, or T.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> The structure of the default XSD cannot be changed.

Before Loading Legacy Data

Before you begin to load the legacy data, you must do the following:

- Install and configure the Pipeline Manager database.
- Prepare the legacy data for mapping. See "Guidelines for Mapping Legacy Data".
- Map the legacy data to Pipeline Manager database format. See "Mapping Legacy Data".

Configuring the Registry File for LoadIfwConfig Utility

The **LoadIfwConfig.reg** file provides database connection information to the **LoadIfwConfig** utility. You can edit this file manually, but it is also updated when you run the **pin_setup** utility.

The **LoadIfwConfig.reg** file is located in *Pipeline_home***/tools/XMLloader.**

Most of the entries are standard connection entries, with these exceptions:

- Use the **LogFileName** entry to specify the file where debug messages are written.

  > **Note**
  >
  > To record debug messages, you must use the **verbose on** command when you run the **LoadIfwConfig** utility.

- Use the **LoadDataFromDB** entry to increase performance. Enabling this entry loads all the pricing data from the database into memory, where the utility can access it faster.

Sample **LoadIfwConfig.reg** file:

```
LOADIFWCONFIG
{
DataPool
   {
      Database
      {
         ModuleName = DbInterface
         Module
        {
            DatabaseName = dduttadb
            UserName     = PIN
            PassWord     = password
            AccessLib    = oci61
         }
      }
   }
XMLSchemaFile = Metadata.xml
LogFileName = LoadIfwConfig.log
LoadDataFromDB = False
}
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

Loading Legacy Data into the Pipeline Manager Database with LoadIfwConfig

To load legacy data into the Pipeline Manager database:

1. Go to *Pipeline_home*/**bin**.
2. Run **LoadIfwConfig** using one of the following commands:
   - **Non-interactive mode**

     ```
     LoadIfwConfig  [-i] input_file  [-I] [-c]
     ```

   - **Interactive mode**

     ```
     LoadIfwConfig  [read input_file] [Insert] [commit]
     ```

> **Note**
>
> The utility must be run from the directory where the XML file is located.

The **LoadIfwConfig** utility uploads the data from the XML file into the Pipeline Manager database and commits the data. If there is failure, the **LoadIfwConfig** utility rolls back the data and notifies you on your computer screen.

Deleting Data from the Pipeline Manager Database with LoadIfwConfig

To delete an object in a table, you must specify the primary field of the object in the input XML file. The object can be deleted only if the primary field has no dependent child objects.

To delete data from the Pipeline Manager database:

1. Go to *Pipeline_home*/**bin**.
2. Run **LoadIfwConfig** using one of the following commands:
   - **Non-interactive mode**

     ```
     LoadIfwConfig  [-i] input_file  [-p] [-c] output_file
     ```

   - **Interactive mode**

     ```
     LoadIfwConfig  [read input_file] [delete] [commit] output_file
     ```

   Only the dependent objects are deleted, and the deleted objects are backed up in the output file.

Deleting Data Sample XML File

Deleting data representation in the XML file

```
<?xml version="1.0" encoding="UTF-8"?>
<IFW xmlns="http://www.portal.com/tools/pricing" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="http://www.portal.com/tools/pricing
IfwConfig.xsd">
<UOM UnitsOfMeasurement="hp1" Name="abc"/></IFW>
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

Updating Data with the LoadIfwConfig Utility

To update data from the Pipeline Manager database:

1. Go to *Pipeline_home*/**bin**.
2. Run **LoadIfwConfig** using one of the following commands:
   - **Non-interactive mode**

     LoadIfwConfig [-i] *input_file* **[-p]** *output_file*
   - **Interactive mode**

     LoadIfwConfig [read *input_file*] **[update] [commit]** *output_file*

Exporting Data from the Database with the LoadIfwConfig Utility

To export data in a table, you must specify the primary field of the object in the input XML file. The utility exports all dependent data. For example, if you specify to export charges, the utility exports charge versions, charge configurations, pricings, impact categories, and time zones.

This sample shows an the contents of an XML file for exporting data:

```
<?xml version="1.0" encoding="UTF-8"?>
<IFW xmlns="http://www.portal.com/tools/pricing" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="http://www.portal.com/tools/pricing
IfwConfig.xsd">
<UOM UnitsOfMeasurement="hp1" Name="abc"/></IFW>
```

To export data from the Pipeline Manager database:

1. Go to *Pipeline_home*/**bin**.
2. Run **LoadIfwConfig** using one of the following commands:
   - **Non-interactive mode**

     LoadIfwConfig [-i] *input_file* **[-f] [-r] [-o]** *output_file*
   - **Interactive mode**

     LoadIfwConfig [read *input_file*] **[fetch] [write] [commit]** *output_file*

The exported objects are written in XML format.

Troubleshooting

Errors during data loading, deleting, and exporting are usually caused by the following:

- The XML data does not comply with the XSD structure.
- You are trying to load data into a table that does not exist.
- You are trying to load data but the prerequisite data does not exist. See "About the Types of Objects to Migrate".
- You are trying to delete data that has other data depending on it.

Copyright © 2025, 2026, Oracle and/or its affiliates.

25

Improving Real-Time Rating Performance

This chapter provides information on evaluating and improving the performance of Oracle Communications Billing and Revenue Management (BRM) real-time rating.

Topics in this document:

- Improving Real-Time Rating Performance
- Improving Performance for Loading Large Price Lists

Improving Real-Time Rating Performance

You can improve real-time rating performance by doing the following:

- Changing the Precision of Rounded and Calculated Values
- Setting the Interval for Checking for Price List Changes
- Setting the Interval for Updating Zone Maps
- Filtering the ERAs Considered During Rating and Discounting
- Enabling and Disabling the Caching of Customized Products
- Configuring the Maximum Number of Products and Discounts Cached

Changing the Precision of Rounded and Calculated Values

To improve performance, you can change the precision of rounded values and of values calculated by real-time rating. You change the precision by adding or modifying entries in the Connection Manager (CM) **pin.conf** file:

- To change the precision of rounded values, add or change the **rating_quantity_rounding_scale** entry. The value of this entry determines the number of digits to the right of the decimal place for rounding quantities. The default is **8**.
- To change the precision of calculated values, add or change the **rating_max_scale** entry. The value of this entry determines the number of digits to the right of the decimal place that are used. The default is **10**.

> **Note**
>
> You must stop and restart the Connection Manager (CM) after you change these values.

Setting the Interval for Checking for Price List Changes

You can set the interval at which BRM checks for changes to the price list. If you change the price list frequently, you may want to use a shorter interval. If your price list is less volatile, you can increase the interval.

Copyright © 2025, 2026, Oracle and/or its affiliates.

To change the interval:

1. Open the CM configuration file (*BRM_home*/**sys/cm/pin.conf**).
2. Edit the following entry:

   ```
   - fm_rate refresh_product_interval 3600
   ```

   The value of this entry determines the interval in seconds. The default is **3600**.
3. Save the file.
4. Stop and restart the CM.

Setting the Interval for Updating Zone Maps

To specify how frequently BRM checks for changes to zone maps and updates them in the database:

1. Open the CM configuration file (*BRM_home*/**sys/cm/pin.conf**).
2. Edit the following entry:

   ```
   - fm_zonemap_pol update_interval 3600
   ```

   The value of this entry determines the interval in seconds. The default is **3600**.
3. Save the file.
4. Stop and restart the CM.

Filtering the ERAs Considered During Rating and Discounting

By default, real-time rating checks for both account-level extended rating attributes (**/profile/acct_extrating** object) and service-level ERAs (**/profile/serv_extrating** object) when it searches for rating and discounting criteria. You can improve real-time rating performance by filtering the types of ERAs that BRM considers when it searches for rating and discounting criteria. For example, you can configure BRM to search for service-level ERAs only or to omit the ERA search altogether.

You can specify the types of ERAs to consider by modifying a field in the **rating** instance of the **/config/business_params** object.

You modify the **/config/business_params** object by using the **pin_bus_params** utility. See "pin_bus_params" in *BRM Developer's Guide*.

To specify the ERA types:

1. Use the following command to create an editable XML file from the **rating** instance of the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsRating bus_params_rating.xml
   ```

   This command creates the XML file named **bus_params_rating.xml.out** in your working directory. If you do not want this file in your working directory, specify the full path as part of the file name.
2. Search the XML file for following line:

   ```
   <EnableEras>serviceAndAccount</EnableEras>
   ```
3. Change **serviceAndAccount** to one of the following:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- **account**: Limits the rating and discounting criteria search to account-level ERAs by retrieving only the **/profile/acct_extrating** object.
- **service**: Limits the rating and discounting criteria search to service-level ERAs by retrieving only the **/profile/serv_extrating** object.
- **disabled**: Omits ERAs from the rating and discounting criteria. Because neither object is retrieved, this option provides the best performance.

> **Note**
>
> BRM uses the XML in this file to overwrite the existing **rating** instance of the **/config/business_params** object. If you delete or modify any other parameters in the file, these changes affect the associated aspects of the BRM rating configuration.

4. Save the file.
5. Change the file name from **bus_params_rating.xml.out** to **bus_params_rating.xml**.
6. Use the following command to load the change into the **/config/business_params** object:

   ```
   pin_bus_params bus_params_rating.xml
   ```

   You should run this command from the *BRM_home***/sys/data/config** directory, which includes support files used by the utility. To run it from a different directory, see "pin_bus_params" in *BRM Developer's Guide*.
7. To verify that all fields are correct, you can display the **/config/business_params** object by using Object Browser in Developer Center or by using the **robj** command with the **testnap** utility.

   For more information, see "Using the testnap Utility to Test BRM" in *BRM Developer's Guide*.
8. Stop and restart the CM.

Enabling and Disabling the Caching of Customized Products

When you use advanced customization to create customized products, BRM uses the customized products for rating. You can control whether customized products are cached for use by the real-time rating engine.

- If you choose not to cache customized products (the default setting), the real-time rating engine retrieves customized product data from the database during rating. This slows rating performance but minimizes the memory impact of customized products.
- If you choose to cache customized products, the CM size grows as customized products are created. Because the products are cached in memory, however, rating performance is increased.

You enable product caching by changing the **EnableTailormadeCache** field in the **rating** instance of the **/config/business_params** object from **disabled** to **enabled**. You can disable caching by changing the field back to **0**.

You modify the **/config/business_params** object by using the **pin_bus_params** utility. See "pin_bus_params" in *BRM Developer's Guide*.

To enable caching of customized products:

Copyright © 2025, 2026, Oracle and/or its affiliates.

1. Use the following command to create an editable XML file from the **rating** instance of the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsRating bus_params_rating.xml
   ```

   This command creates the XML file named **bus_params_rating.xml.out** in your working directory. If you do not want this file in your working directory, specify the full path as part of the file name.

2. In the **bus_params_rating.xml.out** file set **EnableTailormadeCache** to **enabled**:

   ```
   <EnableTailormadeCache>enabled</EnableTailormadeCache>
   ```

   > **Note**
   >
   > BRM uses the XML in this file to overwrite the existing **rating** instance of the **/config/business_params** object. If you delete or modify any other parameters in the file, these changes affect the associated aspects of the BRM rating configuration.

3. Save the file.
4. Change the file name from **bus_params_rating.xml.out** to **bus_params_rating.xml**.
5. Use the following command to load the change into the **/config/business_params** object:

   ```
   pin_bus_params bus_params_rating.xml
   ```

   You should run this command from the *BRM_home*/**sys/data/config** directory, which includes support files used by the utility. To run it from a different directory, see "pin_bus_params" in *BRM Developer's Guide*.

6. To verify that all fields are correct, you can display the **/config/business_params** object by using Object Browser or by using the **robj** command with the **testnap** utility.

   For information on using **testnap**, see "testnap" in *BRM Developer's Guide*.

7. Stop and restart the CM.

Configuring the Maximum Number of Products and Discounts Cached

Products and discounts that are used during the real-time rating process are automatically stored in the CM cache. This improves rating performance, but, over time, it can consume a large amount of memory. To prevent the CM cache from growing too large, you can set a maximum number of products and discounts that can be stored in the CM cache.

- When you set the maximum to a nonzero value, BRM prevents the real-time rating engine from storing more than the specified number of products and discounts in CM cache. When the maximum number is reached, BRM flushes 10% of the products and discounts from cache that have been used the least.

  > **Note**
  >
  > The maximum number of products and discounts that should be stored in CM cache depends on the your business needs.

- When you set the maximum to zero, BRM does not regulate the number of products and discounts stored in the CM cache. This is the default.

Copyright © 2025, 2026, Oracle and/or its affiliates.

You configure the maximum number of products and discounts that can be cached by configuring the **ProductsDiscountsThreshold** field in the **rating** instance of the **/config/business_params** object.

To set a maximum number of products and discounts that can be cached:

1. Use the following command to create an editable XML file from the **rating** instance of the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsRating bus_params_rating.xml
   ```

   This command creates the XML file named **bus_params_rating.xml.out** in your working directory. If you do not want this file in your working directory, specify the full path as part of the file name.

2. Search the XML file for following line:

   ```
   <ProductsDiscountsThreshold>0</ProductsDiscountsThreshold>
   ```

3. Change **0** to the maximum number of products and discounts that you would like stored in cache. The default value of **0** specifies to not regulate the number of products and discounts in the CM cache.

   > **Note**
   >
   > BRM uses the XML in this file to overwrite the existing **rating** instance of the **/config/business_params** object. If you delete or modify any other parameters in the file, these changes affect the associated aspects of the BRM rating configuration.

4. Save the file.
5. Change the file name from **bus_params_rating.xml.out** to **bus_params_rating.xml**.
6. Use the following command to load the change into the **/config/business_params** object:

   ```
   pin_bus_params bus_params_rating.xml
   ```

   You should run this command from the *BRM_home***/sys/data/config** directory, which includes support files used by the utility. To run it from a different directory, see "pin_bus_params" in *BRM Developer's Guide*.

7. To verify that all fields are correct, you can display the **/config/business_params** object by using Object Browser or by using the **robj** command with the **testnap** utility.

   For information on using **testnap**, see "testnap" in *BRM Developer's Guide*.

8. Stop and restart the CM.

Improving Performance for Loading Large Price Lists

If you have a large price list, you can improve performance in the following ways:

- Cache pricing data, such as G/L IDs and resource IDs. In a test environment where you modify your price list often, caching pricing data improves performance because there is no need to load price reference objects every time you commit the price list to the database.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> Pricing data is created every time the CM starts. Whenever the pricing data is changed in the database, the CM must be stopped and restarted to place the new information into the cache.

In a production system where you rarely modify your price list, you do not need to cache pricing data. This reserves the CM cache for other uses and eliminates the need to stop and restart the CM to update the cache if you change the price list.

- Turn off event logging for price list creation events.

> **Note**
>
> When you turn off event logging, BRM still stores audit trail information for products, deals, and plans; however, there will not be an event log of when the price plans were modified and who modified them.

To improve loading performance:

1. Open the CM configuration file (*BRM_home*/**sys/cm/pin.conf**).
2. Specify the cache attributes for storing product provisioning tags. The default entries are usually sufficient.

   ```
   - cm_cache fm_price_prod_provisioning_cache 100, 102400, 13
   ```

3. Specify the cache attributes for storing balance element IDs (BEIDs). The default entries are usually sufficient.

   ```
   - cm_cache fm_price_cache_beid 200, 524288, 32
   ```

4. Specify the cache attributes for storing offer profiles. The default entries are usually sufficient.

   ```
   - cm_cache fm_offer_profile_cache 20, 102400, 13
   ```

   > **Note**
   >
   > For policy-driven charging, the **fm_offer_profile_cache** entry must be present in the **pin.conf** file.

5. Save and exit the file.
6. Go to *BRM_home*/**sys/data/config**.
7. Use the following command to create an editable XML file from the pricing instance of the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsPricing bus_params_pricing.xml
   ```

   This command creates an XML file named **bus_params_pricing.xml.out** in your current directory. If you do not want this file in your current directory, specify the path as part of the file name.

8. In **bus_params_pricing.xml.out**, do the following:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- Set **CacheReferencesAtStart** to **enabled** to cache pricing data or **disabled** to not cache pricing data. The default is **enabled**.

  ```
  <CacheReferencesAtStart>enabled</CacheReferencesAtStart>
  ```

- Set **LogPriceChangeEvent** to **enabled** to log when pricing data is modified or to **disabled** to turn off event logging for pricing data changes. The default is **disabled**.

  ```
  <LogPriceChangeEvent>disabled</LogPriceChangeEvent>
  ```

> ⚠ **Caution**
>
> BRM uses the XML in this file to overwrite the existing instance of the **/config/business_params** object. If you delete or modify any other parameters in the file, these changes affect the associated aspects of the BRM configuration.

9. Save and exit the file.
10. Rename the **bus_params_pricing.xml.out** file to **bus_params_pricing.xml**.
11. Use the following command to load your changes into the **/config/business_params** object:

    ```
    pin_bus_params bus_params_pricing.xml
    ```

    You should run this command from the *BRM_home*/**sys/data/config** directory, which includes support files used by the utility. To run it from a different directory, see "pin_bus_params" in *BRM Developer's Guide*.
12. Read the object with the **testnap** utility or the Object Browser to verify that all fields are correct.

    For general instructions on using **testnap**, see "Using the testnap Utility to Test BRM" in *BRM Developer's Guide*. For information on how to use Object Browser, see "Reading Objects" in *BRM Developer's Guide*.
13. Stop and restart the CM.

    For more information, see "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.

Copyright © 2025, 2026, Oracle and/or its affiliates.

26
Pricing Utilities

This chapter provides reference information for Oracle Communications Billing and Revenue Management (BRM) pricing utilities.

Topics in this document:

- load_brm_pricing
- loadchangesets
- load_event_map
- load_pin_beid
- load_pin_impact_category
- load_pin_rum
- load_pin_spec_rates
- load_pin_sub_bal_contributor
- loadpricelist
- load_usage_map
- pin_apply_promotion
- pin_clean_offer_override

load_brm_pricing

Use this utility to load real-time and pipeline pricing data into the database. It runs the following utilities to load the pricing data:

- **loadpricelist** loads the real-time pricing data into the BRM database. See "loadpricelist" for more information.
- **LoadIfwConfig** loads the pipeline pricing data into the Pipeline Manager database. See "LoadIfwConfig" in *BRM Pipeline Manager Reference* for more information.

Location

BRM_home/**bin**

Syntax

Load data:

```
load_brm_pricing [-v] [-d] [-f] -c Real_time_data_filename -F Pipeline_data_filename
```

Retrieve data:

```
load_brm_pricing [-v] [-d] [-f]  -r Real_time_data_filename -F Pipeline_data_filename
                 [-P] [-D] [-S] [-s ServiceType] [-t ModifiedTime] [-N name]
```

Get help:

Copyright © 2025, 2026, Oracle and/or its affiliates.

```
load_brm_pricing -h
```

Parameters

-v
Displays information about successful or failed processing as the utility runs.

> **Note**
>
> This parameter is always used with other parameters and commands.

-d
Creates a log file for debugging purposes. Use this parameter for troubleshooting when the utility runs with no errors but the event map is not loaded into the database.

-f
Runs the command without prompting for a confirmation.

-c *Real_time_data_filename*
Reads the pricing data from *Real_time_data_filename* and commits it to the BRM database.

> **Note**
>
> When you use the **-c** parameter, it must immediately precede the file name. The **-v**, **-d**, and **-f** parameters can be used in any order, but must precede **-c** *Real_time_data_filename*.

-F *Pipeline_data_filename*
Reads the pricing data from or writes the pricing data to *Pipeline_data_filename*.

- When used with the **-c** parameter, **-F** *Pipeline_data_filename* reads the pricing data from *Pipeline_data_filename* and commits it to the Pipeline Manager database.
- When used with the **-r** parameter, **-F** *Pipeline_data_filename* retrieves the pricing data from the Pipeline Manager database and writes it to *Pipeline_data_filename*.

-r *Real_time_data_filename* [-P] [-D] [-S] [-s *ServiceType*] [-t *ModifiedTime*] [-N *name*]
Retrieves pricing objects from the BRM database and writes the pricing data to *Real_time_data_filename*.

> **Note**
>
> When you use the **-r** parameter, it must immediately precede the file name. The **-v**, **-d**, and **-f** parameters can be used in any order, but must precede **-r** and the file name. The parameters for determining the subset of information can be used in any order *after* the file name.

Use **-r** *Real_time_data_filename* with no additional parameters to retrieve all types of pricing objects from the BRM database.
Use the following parameters with **-r** *Real_time_data_filename* to retrieve only a subset of pricing objects from the BRM database:

Copyright © 2025, 2026, Oracle and/or its affiliates.

- **-P** retrieves only the **/product** objects from the database.
- **-D** retrieves only the **/discount** objects from the database.
- **-S** retrieves only the **/sponsorship** objects from the database.
- **-s** *ServiceType* retrieves objects based on the specified service type. You can list multiple service types by using a comma (**,**) as a delimiter. For example: **-s /service/telco/gsm/telephony, /service/telco/gsm/data**.

> **Note**
>
> Do not use **-s** *ServiceType* with the **-S** parameter.

- **-t** *ModifiedTime* retrieves objects that were modified after the specified timestamp. You specify time using the ISO-8601 standard. Table 26-1 lists the supported formats.

Format	Time Zone
YYYY	Local time of system used to run **load_brm_pricing**.
YYYY-MM	Local time of system used to run **load_brm_pricing**.
YYYY-MM-DD	Local time of system used to run **load_brm_pricing**.
*YYYY-MM-DD*T*hh:mm*Z	UTC
*YYYY-MM-DD*T*hh:mm:ss*Z	UTC
*YYYY-MM-DD*T*hh:mm*[+|-]*TZD*	*TZD* is the time zone relative to UTC. You can use a negative or positive offset from 00:00 to 12:00 (for example, **-05:00** or **+10:00**).
YYYY-MM-DDThh:mm:ss[+|-]*TZD*	*TZD* is the time zone relative to UTC. You can use a negative or positive offset from 00:00 to 12:00 (for example, **-05:00** or **+10:00**).

- **-N** *name* retrieves objects based on the specified charge offer, discount offer, or sponsorship.

> **Note**
>
> You can use an asterisk (*) as a wildcard character to match zero or more characters in the name (for example, **-N** ***GSM***). Use **-N** with **-P**, **-D**, or **-S** to indicate the object type to be retrieved.

-h
Displays the syntax for the **load_brm_pricing** utility.

Results

The **load_brm_pricing** utility notifies you when it successfully completes a command.

If the utility does not notify you that it was successful, look in the utility log file to find any errors. The log file is either in the directory from which the utility was started or in a directory specified in the utility configuration file.

Copyright © 2025, 2026, Oracle and/or its affiliates.

If an error occurs, enter the command again. The **load_brm_pricing** utility does not commit the pricing data to the database until it completes the command.

loadchangesets

Use this utility to import or export pipeline pricing data change sets. You can use the utility in interactive mode or non-interactive mode. In interactive mode, you enter single-word commands to perform individual actions.

You specify the BRM server to export from and the server to import to in a configuration file.

Location

*Pricing_Center_home***/lib**

Syntax: Non-Interactive Mode

```
loadchangesets    [-v] [-fi change_set_file]
                  [-v] [-fx change_set_file]
```

Parameters: Non-Interactive Mode

-v
Displays information about successful or failed processing as the utility runs.

-f *change_set_file*
Mandatory parameter that must precede **-i** and **-x**. Forces processing without prompting for confirmation.

-i *change_set_file*
When used with the **-f** parameter, imports change sets from *change_set_file* to the database. The file name must include the extension **.exp**.
When used alone, switches the utility to interactive mode. See "Syntax: Interactive Mode".

-x
When used with the **-f** parameter, exports the change sets from the database to *change_set_file*. Must be preceded with the **-f** parameter. The file name must include the extension **.exp**.

Syntax: Interactive Mode

```
loadchangesets -i

[read change_set_file]
[export]
[write change_set_file]
[import]
[help]
[quit]
[list]
```

Parameters: Interactive Mode

-i
Switches the utility to interactive mode. After entering interactive mode, commands are single words without the utility name.

Copyright © 2025, 2026, Oracle and/or its affiliates.

export
Exports change sets from the database into memory.

write *change_set_file*
Writes the change sets stored in memory to *change_set_file*. Be sure to include the file name extension **.exp**.

read *change_set_file*
Reads change sets from *change_set_file* and stores them in memory. The file name must include the extension **.exp**.

import
Imports the change sets stored in memory to the database.

list
Lists the change sets currently stored in memory.

help
Displays help the syntax for the utility.

quit
Exits the utility.

Results: Export

Exported files are initially created in the *Pricing_Center_home***/export** directory. The status of the exported change sets changes to **Exported**. If all change sets are exported successfully, the file moves automatically into the **/export/done** directory.

If there are any errors during export, the entire transaction is rolled back. In addition, the status of the change sets is automatically reset from **Exported** to **In Progress**.

Results: Import

The import process generates a log file called **import.log.0**. The log file contains information about validation errors, change sets that have passed validation, and change sets that have been imported successfully.

If all change sets are imported without any errors, the transaction is committed. If there are any errors, the entire transaction is rolled back.

load_event_map

Use this utility to load the event maps into the BRM database. You define the event maps in the *BRM_home***/sys/data/pricing/example/pin_event_map** file.

For more information about event maps, see "Mapping Events and Services".

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> - The **load_event_map** utility overwrites the entire event map. If you are updating the event map, you cannot load new mappings only. You load the entire event map each time you run the **load_event_map** utility. The **load_event_map** utility does not restrict to load any particular event type with a service. However, if the service types and event pairs are duplicated, the **load_event_map** utility reports an error.
> - To connect to the BRM database, the **load_event_map** utility needs a configuration file in the directory from which you run the utility. See "Connecting BRM Utilities" in *BRM System Administrator's Guide*.

Location

*BRM_home***/bin**

Syntax

```
load_event_map [-d] [-v] pin_event_map_file
```

Parameters

-d
Logs messages to the **default.pinlog** file in the current directory, or in a directory specified in the utility configuration file. Use this parameter for troubleshooting when the utility runs with no errors, but the event map is not loaded into the database.

-v
Displays information about successful or failed processing as the utility runs.

> **Note**
>
> This parameter is always used with other parameters and commands. It is not position dependent. For example, you can enter **-v** at the beginning or end of a command to initiate the verbose parameter. To redirect the output to a log file, use the following syntax with the verbose parameter. Replace *filename***.log** with the name of the log file:
> **load_event_map** *other_parameter* **–v >** *filename***.log**

pin_event_map_file
The name and location of the file that defines event maps. The default is *BRM_home***/sys/data/pricing/example/pin_event_map**.
If you run the command from a different directory from the one in which *pin_event_map_file* is located, you must include the entire path for the file.

Results

The **load_event_map** utility notifies you only if it encounters errors. Look in the **default.pinlog** file for errors. This file is either in the directory from which the utility was started, or in a directory specified in the utility configuration file.

Copyright © 2025, 2026, Oracle and/or its affiliates.

load_pin_beid

Use this utility to load resources into the BRM database. You define the resources in the *BRM_home***/sys/data/pricing/example/pin_beid** file.

> **Note**
>
> It is easier to use the Resource Editor to define resources.

For more information about defining resources, see "About Resources".

> **Note**
>
> - The **load_pin_beid** utility overwrites the existing resources. If you are updating resources, you cannot load new resources only. You load complete sets of resources each time you run the **load_pin_beid** utility.
> - When you add or edit resources by using the Resource Editor, the Resource Editor does not update the **pin_beid** file. Therefore, if you want the **pin_beid** file to include all of your resources, you must edit it by hand.
> - To connect to the BRM database, the **load_pin_beid** utility needs a configuration file in the directory from which you run the utility. See "Connecting BRM Utilities" in *BRM System Administrator's Guide*.

Location

*BRM_home***/bin**

Syntax

```
load_pin_beid [-d] [-v] pin_beid_file
```

Parameters

-d
Sends all flists used in creating the resources to a log file. Use this parameter for debugging when the utility appears to have run with no errors, but the resources do not appear in the database.

-v
Displays information about successful or failed processing as the utility runs.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> This parameter is always used with other parameters and commands. It is not position dependent. For example, you can enter **-v** at the beginning or end of a command to initiate the verbose parameter. To redirect the output to a log file, use the following syntax with the verbose parameter. Replace *filename***.log** with the name of the log file:
> **load_pin_beid** *other_parameter* **–v >** *filename***.log**

pin_beid_file
The name and location of the file that defines resources. The default is *BRM_home***/sys/data/pricing/example/pin_beid**.
If you copy the **pin_beid** file to the same directory from which you run the **load_pin_beid** utility, you do not have to specify either the path or the file name.

Results

The **load_pin_beid** utility notifies you only if it encounters errors.

> **Note**
>
> You must restart the Connection Manager to make the new resources available.

Guideline for Defining Resource Name

When you define the resource name in the **pin_beid** file for the **load_pin_beid** utility, and if you use numeric values in the resource name, do not separate the numeric values at the end with whitespace. If whitespace is present, the numeric values will correspond to the resource's validity in days.

For example, consider the following resource definition in the **pin_beid** file:

1000111 0 0 0.000000 0.000000 0.000000 GPRS VOLUM * 2 0 0 10 MB FREE FEB 2010

This sets the **PIN_FLD_VALIDITY_IN_DAYS** field to 2010.

If you use '10 MB FREE FEB2010' as the resource name instead of '10 MB FREE FEB 2010', the **PIN_FLD_VALIDITY_IN_DAYS** field will not be set to 2010, and it will remain as the resource name.

load_pin_impact_category

Use this utility to load impact categories into the BRM database. You define the impact categories for the rate plan selector charges in the *BRM_home***/sys/data/config/pin_impact_category** file.

For information about impact categories, see "About Impact Categories".

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> - The **load_pin_impact_category** overwrites the existing impact categories. If you are updating a set of impact categories, you cannot load new impact categories only. You load complete sets of impact categories each time you run the **load_pin_impact_category** utility.
> - To connect to the BRM database, the **load_pin_impact_category** utility needs a configuration file in the directory from which you run the utility. See "Connecting BRM Utilities" in *BRM System Administrator's Guide*.

Location

*BRM_home***/bin**

Syntax

```
load_pin_impact_category pin_impact_category_file
```

Parameters

pin_impact_category_file
The name and location of the file that defines the impact categories. The default is *BRM_home***/sys/data/config/pin_impact_category**.
If you copy the **pin_impact_category** file to the same directory from which you run the **load_pin_impact_category** utility, you do not have to specify either the path or the file name.

Results

The **load_pin_impact_category** utility notifies you when it successfully creates the **/config/impact_category** storable object.

If the **load_pin_impact_category** utility does not notify you that it was successful, look in the **default.pinlog** file to find any errors. This file is either in the directory from which the utility was started or in a directory specified in the utility configuration file.

> **Note**
>
> You must restart the Connection Manager to make the new impact categories available.

load_pin_rum

Use this utility to load ratable usage metrics (RUMs) into the **/config/rum** storable object in the BRM database. You define RUMs in the *BRM_home***/sys/data/pricing/example/pin_rum** file.

For information about RUMs, see "Real-Time Rating Based on Multiple RUMs".

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> - The **load_pin_rum** utility overwrites the existing RUMs. If you are updating a set of RUMs, you cannot load new RUMs only. You load complete sets of RUMs each time you run the **load_pin_rum** utility.
> - To connect to the BRM database, the **load_pin_rum** utility needs a configuration file in the directory from which you run the utility. See "Connecting BRM Utilities" in *BRM System Administrator's Guide*.

Location

*BRM_home***/bin**

Syntax

```
load_pin_rum [-d] [-v] pin_rum_file
```

Parameters

-d
Creates a log file for debugging purposes. Use this parameter for debugging when the utility appears to have run with no errors, but the RUMs do not appear in the database.

-v
Displays information about successful or failed processing as the utility runs.

> **Note**
>
> This parameter is always used with other parameters and commands. It is not position dependent. For example, you can enter **-v** at the beginning or end of a command to initiate the verbose parameter. To redirect the output to a log file, use the following syntax with the verbose parameter. Replace *filename***.log** with the name of the log file:
> **load_pin_rum** *other_parameter* **–v >** *filename***.log**

pin_rum_file
The name and location of the file that defines RUMs. The default is *BRM_home***/sys/data/pricing/example/pin_rum**.
If you copy the **pin_rum** file to the same directory from which you run the **load_pin_rum** utility, you do not have to specify either the path or the file name.

Results

The **load_pin_rum** utility notifies you only if it encounters errors.

> **Note**
>
> You must restart the Connection Manager to make the new RUMs available.

Copyright © 2025, 2026, Oracle and/or its affiliates.

load_pin_spec_rates

Use this utility to set up customized rating by loading the contents of the **pin_spec_rates** file into the BRM database.

For information about customizing rating, see *BRM Opcode Guide*.

> **Note**
>
> - The **load_pin_spec_rates** overwrites the existing setup for administrative events charges. If you are updating a set of administrative events charges, you cannot load new charges only. You load complete sets of charges each time you run the **load_pin_spec_rates** utility.
> - To connect to the BRM database, the **load_pin_spec_rates** utility needs a configuration file in the directory from which you run the utility. See "Connecting BRM Utilities" in *BRM System Administrator's Guide*.

Location

*BRM_home***/bin**

Syntax

```
load_pin_spec_rates pin_spec_rates_file
```

Parameters

pin_spec_rates_file
The name and location of the file that maps opcodes to event types to be rated. The default is *BRM_home***/sys/data/config/pin_spec_rates**.
If you copy the **pin_spec_rates** file to the same directory from which you run the **load_pin_spec_rates** utility, you do not have to specify either the path or the file name.

Results

If the **load_pin_spec_rates** utility does not notify you that it was successful, look in the **default.pinlog** file to find any errors. This file is either in the directory from which the utility was started or in a directory specified in the utility configuration file.

> **Note**
>
> You must restart the Connection Manager for the new administrative event charges to take effect.

load_pin_sub_bal_contributor

Use this utility to load the configuration for contributor-based sub-balances into the BRM database. You define the sub-balance configuration in the *BRM_home***/sys/data/pricing/example/pin_sub_bal_contributor** file.

For information about contributor-based sub-balances, see "Managing Sub-Balances".

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> - The **load_pin_sub_bal_contributor** utility overwrites the existing sub-balance configurations. If you are updating a set of sub-balance configurations, you cannot load new configurations only. You load complete sets of sub-balance configurations each time you run the **load_pin_sub_bal_contributor** utility.
> - To connect to the BRM database, the **load_pin_sub_bal_contributor** utility needs a configuration file in the directory from which you run the utility. See "Connecting BRM Utilities" in *BRM System Administrator's Guide*.
> - The resources specified in the **pin_sub_bal_contributor** file must exist in the BRM database before running the **load_pin_sub_bal_contributor** utility. This includes any aggregation counters used to track consumption of resources that are shared among several accounts. Therefore, you must first load the **pin_beid** file by running the **load_pin_beid** utility. See "load_pin_beid".

Location

*BRM_home***/bin**

Syntax

```
load_pin_sub_bal_contributor [-d] [-v] pin_sub_bal_contributor_file
```

Parameters

-d
Creates a log file for debugging purposes. Use this parameter for debugging when the utility appears to have run with no errors, but the sub-balance configurations have not been loaded into the database.

-v
Displays information about successful or failed processing as the utility runs.

> **Note**
>
> This parameter is always used with other parameters and commands. It is not position dependent. For example, you can enter **-v** at the beginning or end of a command to initiate the verbose parameter. To redirect the output to a log file, use the following syntax with the verbose parameter. Replace *filename***.log** with the name of the log file:
> **load_pin_sub_bal_contributor** *other_parameter* **–v >** *filename***.log**

pin_sub_bal_contributor_file
The name and location of the file that defines the configuration for contributor-based sub-balances. The default **pin_sub_bal_contributor** file is in *BRM_home***/sys/data/pricing/sample**.
If you copy the **pin_sub_bal_contributor** file to the directory from which you run the **load_pin_sub_bal_contributor** utility, you do not have to specify either the path or the file name.

Copyright © 2025, 2026, Oracle and/or its affiliates.

Results

The **load_pin_sub_bal_contributor** utility notifies you when it successfully creates the **/config/sub_bal_contributor** object.

If the **load_pin_sub_bal_contributor** utility does not notify you that it was successful, look in the **default.pinlog** file to find any errors. This log file is either in the directory from which the utility was started or in a directory specified in the utility configuration file.

> **Note**
>
> You must restart the Connection Manager to make the new sub-balance configurations available.

loadpricelist

Use this utility to prepare and commit price list information, such as products, deals, plans, offer profiles, and plan lists, to your BRM database. You define the price list or offer profile configurations by using the **price_list.xsd** file in the *BRM_home*/**PricingCenter** directory.

The **loadpricelist** utility is the XML Pricing Interface.

> **Note**
>
> - Do not use the **loadpricelist** utility to assign custom rateable usage metrics (RUMs) to fold events in any rate plans. Fold events cannot use custom RUMs; therefore, products configured with them are rated incorrectly.
> - To connect to the BRM database, the CLASSPATH should point to an **Infranet.properties** file with the correct login information for the **loadpricelist** utility. The default **Infranet.properties** file is in the **/common** directory. To put the **Infranet.properties** file in the local directory with the **loadpricelist** utility, add period-slash (**./**) to the beginning of the CLASSPATH.

For information, see "Using the XML Pricing Interface to Create a Price List ".

Location

BRM_home/**bin**

Syntax: Non-Interactive Mode

Copy:

```
loadpricelist [-v] [-d] [-f] -c PriceListFile
```

Purge:

```
loadpricelist [-v] [-d] [-f] -p
```

Retrieve:

```
loadpricelist [-v] [-d] [-f] -r PriceListFile [-P] [-D] [-S] [-z] [-s ServiceType] [-t
ModifiedTime] [-N Name]
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

Get help:

```
loadpricelist -h
```

Parameters: Non-Interactive Mode

-v
Displays information about successful or failed processing as the utility runs.

> **Note**
>
> This parameter is always used with other parameters and commands.

-d
Creates a log file for debugging purposes. Use this parameter for debugging when the utility appears to have run with no errors but the data has not been loaded into the database.

-f
Runs the command without prompting for a confirmation.

-c *PriceListFile*
Reads the price list information from *PriceListFile* and commits it to the BRM database.

> **Note**
>
> - When you use the **-c** parameter, it must immediately precede the file name. The **-v**, **-d**, and **-f** parameters can be used in any order, but must precede **-c** and the file name.
> - *PriceListFile* must be in the same directory as the **price_list.xsd** file. If they are not in the same directory, edit the **noNamespaceSchemaLocation** entry in *PriceListFile* to include the full path to **price_list.xsd**. By default, **price_list.xsd** is in the *BRM_home***PricingCenter** directory. For example, if the XSD file is located in **C:\pricelists**, change the entry to:
>
> ```
> xsi:noNamespaceSchemaLocation="C:\pricelists\price_list.xsd"
> ```

-p
Purges pricing objects from the BRM database.

-r *PriceListFile* [-P] [-D] [-S] [-z] [-s *ServiceType*] [-t *ModifiedTime*] [-N *Name*]
Retrieves pricing objects from the BRM database and writes them to *PriceListFile*. When the **infranet.batchsize** entry is present in the **loadpricelist Infranet.properties** file, the utility retrieves objects in batches based on the value of the entry. See "Downloading Price Lists in Batches".
Use **-r** *PriceListFile* with no additional parameters to retrieve all types of pricing objects from the BRM database.
Use the following parameters with **-r** *PriceListFile* to retrieve only a subset of pricing objects from the BRM database:

- **-P** retrieves only the **/product** objects from the database.
- **-D** retrieves only the **/discount** objects from the database.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- **-S** retrieves only the **/sponsorship** objects from the database.
- **-z** retrieves only the **/zonemap** objects from the database.
- **-s** *ServiceType* retrieves objects based on the specified service type. You can list multiple service types by using a comma (**,**) as a delimiter. For example: **-s /service/telco/gsm/telephony, /service/telco/gsm/data**.

> **Note**
>
> Do not use **-s** *ServiceType* with the **-S** parameter.

- **-N** *Name* retrieves the product, discount, or sponsorship based on its name. The name can contain * to match any character zero or more times anywhere in the name.
- **-t** *ModifiedTime* retrieves objects that were modified after the specified timestamp. You specify time using the ISO-8601 standard. Table 26-2 lists the supported formats:

Format	Time Zone
YYYY	Local time of system used to run **loadpricelist**.
YYYY-MM	Local time of system used to run **loadpricelist**.
YYYY-MM-DD	Local time of system used to run **loadpricelist**.
*YYYY-MM-DD*T*hh*:*mm*Z	UTC
*YYYY-MM-DD*T*hh*:*mm*:*ss*Z	UTC
*YYYY-MM-DD*T*hh*:*mm*[+|-]*TZD*	*TZD* is the time zone relative to UTC. You can use a negative or positive offset from 00:00 to 12:00 (for example, **-05:00** or **+10:00**).
*YYYY-MM-DD*T*hh:mm:ss*[+|-]*TZD*	*TZD* is the time zone relative to UTC. You can use a negative or positive offset from 00:00 to 12:00 (for example, **-05:00** or **+10:00**).

> **Note**
>
> When you use the **-r** parameter, it must immediately precede the file name. The **-v**, **-d**, and **-f** parameters can be used in any order, but must precede **-r** and the file name. The parameters for determining the subset of information can be used in any order *after* the file name.

-h
Displays the syntax for the **loadpricelist** utility.

Syntax: Interactive Mode

```
loadpricelist    read PriceListFile

                 write
                   PriceListFile [planlist [type] | plan | subscriptionterm |
                   | bestpricing | deal | product | discount | sponsorship |
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

```
  | offerprofile] [PriceObjectName]

write
  PriceListFile [dependency | transition]

delete
  [planlist [type] | plan | bestpricing | deal | subscriptionterm |
  | product | discount | sponsorship |
  | offerprofile] [PriceObjectName]

delete
  [dependency | transition]

retrieve
  [-s ServiceType] [-t ModifiedTime] [product]
  [discount] [sponsorship]

commit
purge
help
quit
verbose [on | off | debug]
list
```

Parameters: Interactive Mode

read *PriceListFile*
Reads the price list information from *PriceListFile* and stores it in internal memory as an flist.

> **Note**
>
> - **loadpricelist** maintains only one flist. When you load the price list information from your database, **loadpricelist** removes the current flist and creates a new flist.
> - *PriceListFile* must be in the same directory as the **price_list.xsd** file. If they are not in the same directory, edit the **noNamespaceSchemaLocation** entry in *PriceListFile* to include the full path to **price_list.xsd**. By default, **price_list.xsd** is in the *BRM_home***PricingCenter** directory. For example, if the XSD file is located in **C:\pricelists**, change the entry to: **xsi:noNamespaceSchemaLocation="C:\pricelists\price_list.xsd"**

write
Writes the price list information from the in-memory internal flist into *PriceListFile*.

```
write PriceListFile [planlist [type] | plan | bestpricing | deal | subscriptionterm |
product | discount | sponsorship | offerprofile] [PriceObjectName]

write PriceListFile [dependency | transition]
```

Use **write** *PriceListFile* without any other parameters to write *all* pricing objects from the in-memory internal flist into *PriceListFile*.
Use the following parameters to write only a subset of pricing objects from the in-memory internal flist into *PriceListFile*:

- **planlist** [*type*] specifies to write only plan list information into *PriceListFile*. You can optionally specify to write only a plan list object of a particular type into *PriceListFile*.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- **plan** specifies to write only the **/plan** objects into *PriceListFile*.
- **bestpricing** specifies to write only the **/bestpricing** objects into *PriceListFile*.
- **deal** specifies to write only the **/deal** objects into *PriceListFile*.
- **subscriptionterm** specifies to write only the **/subscription_terms** objects into *PriceListFile*.
- **product** specifies to write only the **/product** objects into *PriceListFile*.
- **discount** specifies to write only the **/discount** objects into *PriceListFile*.
- **sponsorship** specifies to write only the **/sponsorship** objects into *PriceListFile*.
- **offerprofile** specifies to write only the **/offer_profile** objects into *PriceListFile*.
- *PriceObject* specifies to write only the specified pricing object into *PriceListFile*.
- **dependency** specifies to write only **/dependency** objects into *PriceListFile*.
- **transition** specifies to write only **/transition information** objects into *PriceListFile*.

delete
Deletes in-memory price list information from the BRM database.

```
delete [planlist [type] | plan | bestpricing | deal | subscriptionterms | product |
discount | sponsorship | offerprofile]
       [PriceObject]

delete [dependency | transition]
```

Use **delete** without any other parameters to delete *all* in-memory pricing information from the database.
Use the following parameters to delete only a subset of in-memory pricing information from the database:

- **planlist** [*type*] specifies to delete the plan list information of a particular type.
- **plan** specifies to delete only the **/plan** objects from the database.
- **bestpricing** specifies to delete only the **/bestpricing** objects from the database.
- **deal** specifies to delete only the **/deal** objects from the database.
- **subscriptionterms** specifies to delete only the **/subscription_terms** objects from the database.
- **product** specifies to delete only the **/product** objects and all related product information, such as the rate plan, rate, rate tier, and balance impact, from the database.
- **discount** specifies to delete only the **/discount** objects from the database.
- **sponsorship** specifies to delete only the **/sponsorship** objects from the database.
- **offerprofile** specifies to delete only the **/offer_profile** objects from the database.
- *PriceObject* specifies to delete only the specified pricing object from the database.
- **dependency** specifies to delete only **/dependency** objects.
- **transition** specifies to delete only **/transition** objects.

retrieve
Retrieves the specified price list objects from the BRM database and stores it in internal memory as an flist.

```
retrieve  [-s ServiceType] [-t ModifiedTime] [product] [discount] [sponsorship]
```


Copyright © 2025, 2026, Oracle and/or its affiliates.

Use **retrieve** without any other parameters to retrieve *all* pricing objects from the BRM database.
Use the following parameters to retrieve only a subset of pricing objects from the BRM database:

- **-s** *ServiceType* retrieves objects based on the specified service type. Multiple service types are delimited with a comma (**,**).

  Do not use this parameter with the **sponsorship** parameter.

- **-t** *ModifiedTime* retrieves objects that were modified after the specified time. You specify time using the ISO-8601 standard. Table 26-3 lists the supported formats.

Format	Time Zone
YYYY	Local time of system used to run **loadpricelist**.
YYYY-MM	Local time of system used to run **loadpricelist**.
YYYY-MM-DD	Local time of system used to run **loadpricelist**.
*YYYY-MM-DD*T*hh*:*mm*Z	UTC
*YYYY-MM-DD*T*hh*:*mm*:*ss*Z	UTC
*YYYY-MM-DD*T*hh*:*mm*[+|-]*TZD*	*TZD* is the time zone relative to UTC. You can use a negative or positive offset from 00:00 to 12:00 (for example, **-05:00** or **+10:00**).
*YYYY-MM-DD*T*hh:mm:ss*[+|-]*TZD*	*TZD* is the time zone relative to UTC. You can use a negative or positive offset from 00:00 to 12:00 (for example, **-05:00** or **+10:00**).

- **product** retrieves only the **/product** objects from the database.
- **discount** retrieves only the **/discount** objects from the database.
- **sponsorship** retrieves only the **/sponsorship** objects from the database.

commit
Commits the pricing information to the database.

purge
Removes all price objects from the database.

help
Displays the syntax for the **loadpricelist** utility.

quit
Exits the utility.

verbose on | off | debug
Sets verbose information:

- **verbose on** displays the status of the command most recently run.
- **verbose off** displays the status only if there is an error.

Copyright © 2025, 2026, Oracle and/or its affiliates.

- **verbose debug** logs the status of the command to the log file specified in the **Infranet.properties** file. The default log file is **javapcm.log**.

list
Prints to the screen all price list information currently stored in internal memory. Because the in-memory price list is in flist format, the utility displays the price list in flist format.

Results

The **loadpricelist** utility notifies you when it successfully completes a command.

If the utility does not notify you that it was successful, look in the **javapcm.log** file to find any errors. This log file is either in the directory from which the utility was started or in a directory specified in the Java property file.

If an error occurs, enter the command again. The **loadpricelist** utility does not save changes until it completes the command.

load_usage_map

Use this utility to load the usage map into the **/config/usage_map/system** storable object in the BRM database. You define the usage map in the *BRM_home***/sys/data/pricing/example/pin_usage_map** file.

For more information, see "Mapping Event Types to RUMs".

> **Note**
>
> - The **load_usage_map** overwrites the existing usage maps. If you are updating a set of usage maps, you cannot load new usage maps only. You load complete sets of usage maps each time you run the **load_usage_map** utility.
> - To connect to the BRM database, the **load_usage_map** utility needs a configuration file in the directory from which you run the utility. See "Connecting BRM Utilities" in *BRM System Administrator's Guide*.

Location

*BRM_home***/bin**

Syntax

```
load_usage_map [-d] [-v] pin_usage_map_file
```

Parameters

-d
Creates a log file for debugging purposes. Use this parameter for debugging when the utility appears to have run with no errors, but the usage maps do not appear in the database.

-v
Displays information about successful or failed processing as the utility runs.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> This parameter is always used with other parameters and commands. It is not position dependent. For example, you can enter **-v** at the beginning or end of a command to initiate the verbose parameter. To redirect the output to a log file, use the following syntax with the verbose parameter. Replace *filename***.log** with the name of the log file:
> **load_usage_map** *other_parameter* **–v >** *filename***.log**

pin_usage_map_file
The name and location of the file that defines usage maps. The default is *BRM_home***/sys/data/pricing/example/pin_usage_map**.
If you copy the **pin_usage_map** file to the same directory from which you run the **load_usage_map** utility, you do not have to specify either the path or the file name.

Results

The **load_usage_map** utility notifies you only if it encounters errors.

pin_apply_promotion

Use this utility to apply promotions, such as free data or minutes, to accounts based on special dates. This utility:

- Searches through all **/profile/specialdates** objects for a PIN_FLD_NEXT_DATE that is less than or equal to the current date and then applies the promotion.
- Sets the PIN_FLD_NEXT_DATE field in the **/profile/specialdates** object to the next occurrence of the special date.

For more information, see "Working with Promotions".

> **Note**
>
> - To access the BRM database, this utility requires a configuration file (*BRM_home***/apps/pin_billd/pin.conf**) in the directory from which you run the utility. See "Connecting BRM Utilities" in *BRM System Administrator's Guide*.
> - You can tune the performance of this multithreaded application (MTA) utility by using the MTA configuration file entries. See "Improving Billing Performance" in *BRM System Administrator's Guide*.

Location

*BRM_home***/bin**

Syntax

```
pin_apply_promotion [-verbose] [-help]
```

Parameters

-verbose
Displays information about successful or failed processing as the utility runs.

Copyright © 2025, 2026, Oracle and/or its affiliates.

-help
Displays the syntax for the **pin_apply_promotion** utility.

Results

The **pin_apply_promotion** utility writes the results to a log file, which is located in the path you specified in the utility's **pin.conf** configuration file.

pin_clean_offer_override

Use this utility to remove expired date range and pricing tag combinations from the ***/offering_override_values*** object.

For more information, see "Dynamically Changing One-Time and Recurring Fees Based On the Date".

> **Note**
>
> - To access the BRM database, this utility requires a configuration file (*BRM_home***/apps/pin_subscription/pin.conf**) in the directory from which you run the utility. See "Connecting BRM Utilities" in *BRM System Administrator's Guide*.
> - You can tune the performance of this multithreaded application (MTA) utility by using the MTA configuration file entries. See "Improving Billing Performance" in *BRM System Administrator's Guide*.

Location

*BRM_home***/bin**

Syntax

```
pin_clean_offer_override [-end date] [-event eventType] [-verbose] [-help]
```

Parameters

-end *date*
Removes all date range and pricing tag combinations that expired prior to the specified date. The date must be in the format *MM*/*DD*/*YYYY*. If the date is not specified, the utility uses the current date.

-event *eventType*
Removes all date range and pricing tag combinations associated with the specified event type.

-verbose
Displays information about successful or failed processing as the utility runs.

Copyright © 2025, 2026, Oracle and/or its affiliates.

> **Note**
>
> This parameter is always used with other parameters and commands. It is not position dependent. For example, you can enter **-v** at the beginning or end of a command to initiate the verbose parameter. To redirect the output to a log file, use the following syntax with the verbose parameter. Replace *filename***.log** with the name of the log file:
> **pin_clean_offer_override** *other_parameter* **–v >** *filename***.log**

-help
Displays the syntax for the utility.

Results

The **pin_clean_offer_override** utility notifies you only if it encounters errors. Look in the **default.pinlog** file for errors. This file is either in the directory from which the utility was started or in a directory specified in the utility configuration file.

Copyright © 2025, 2026, Oracle and/or its affiliates.