Oracle® Communications Billing and Revenue Management

PDC Creating Product Offerings



Release 16.26.0
G54483-01
October 2026

ORACLE

Oracle Communications Billing and Revenue Management PDC Creating Product Offerings, Release 16.26.0

G54483-01

Copyright © 2017, 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.

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Oracle customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

Contents

About This Content

1 About Creating Product Offerings

About Pricing Components 1
About Pricing Setup Components 3
Prerequisites for Creating Product Offerings 5
How PDC Tracks and Publishes Component Changes 5
- How PDC Creates and Activates Changesets 5
- How PDC Publishes a Changeset 6
- How to Import, Export, and Share Changesets 6

Making Changes to Your Product Offerings 6
- Deleting Charge Offers 6
- Logging Changes to Product Offerings 6

About Product Offering Ownership 7

Part I Creating Pricing Setup Components

2 Creating Pricing Setup Components

About Setup Components 1
Configuring Setup Components 2
Configuring Ratable Usage Metrics (RUMs) 5
Setting Up Service and Event Definitions 6
- Using Service and Account Attributes with ECE for Usage Rating 7

Setting Up the Service-Event Map 8
Configuring Balance Elements 9
- Adding Rounding Rules to Balance Elements 10
- Enabling Single Currency Graph 11

Configuring Price Tags 13
About Offer Profiles 14

Copyright © 2017, 2026, Oracle and/or its affiliates.

3 Synchronizing Pricing Setup Components

About Synchronizing Setup Components 1
About the SyncPDC Utility 2
- Prerequisites for Running SyncPDC 2
- About Configuring SyncPDC 2
- Running SyncPDC 7
- Scheduling Synchronization 8
  - Running SyncPDC Immediately 8
  - Running SyncPDC at a Scheduled Time 8
  - Running SyncPDC Immediately and at a Scheduled Time 9
  - Running SyncPDC at Recurring Scheduled Time 9
- Synchronizing BRM Data with PDC to Use ECE for Usage Rating 10
- Stopping SyncPDC 11
- Generating Synchronization Reports 11
- Handling Synchronization Failures 12
  - Reprocessing a Failed Synchronization 13

Changing Display Names 13

4 Configuring Deliverables

About Creating Deliverables 1
Configuring Deliverables 2
- Configuring Deliverables for Physical Goods 5
- Configuring Deliverables for One-Time Activities 6
- Configuring Deliverables for Metered Services 7
- Configuring Deliverables for Continuous Services 9

Loading Deliverables into the PDC Database 10

5 Configuring Policy Specifications

About Configuring Policy Specifications 1
- Configuring Policy Specifications 1

6 Configuring Item Type Selectors

About Configuring Item Type Selectors 1
Loading the OOB_ItemTypeSelector.XML File 4
Configuring Item Type Selectors 5

7 Configuring Value Maps

About Value Maps 1

Copyright © 2017, 2026, Oracle and/or its affiliates.

Loading the Sample Value Map into the Database 2
Configuring Value Maps Using SampleValueMap.XML 2

Part II Creating Pricing Components

8 Configuring Charge Offers

About Charge Offers and Charges 1
Configuring Charge Offer Usage and Ownership 3
Applying Recurring Charges on a Specific DOM 5
Configuring Charge Offers and Revenue Recognition 5
Restricting the End Time of Granted Balances That Start on First Usage 6
Configuring How Pricing Configurations for Date Ranges are Applied 7
Charge Offers and General Ledger 8
Configuring Full Day Proration 8
Configuring Deferred Billing in Charges 8

9 Configuring Charges in Charge Offers

About Charges 1
Specifying When a Charge is Effective 2
Specifying Whether to Charge Inactive, Canceled, or SuspendedActive Accounts 3
Configuring Taxation in Charges 4
Configuring In-Advance Billing in Charges 5
Configuring Cycle Alignment for Recurring Charges 5
About Prorating Recurring Charges and Rollovers 5
About Creating Charge Offers with Over 500 Charges 6

10 Configuring Pricing in Charge Offers

Configuring Pricing in Charges 1
Configuring Validity Periods for Noncurrency Credit Balances 2
Example Configuration for Setting Validity Periods for Noncurrency Balances Based on First Usage Date 3
Granting Noncurrency Balances in Increments 3
Specifying Rounding in Balance Impacts 5
Specifying if a Balance Impact is Discountable 5
About Minimum Charges 6
About Conditional Balance Impacts 6
About Prorating Charges 8
Configuring Dynamic Charging 8

Copyright © 2017, 2026, Oracle and/or its affiliates.

11 Pricing Configurations in Charge Offers

Configuring Pricing Based on Quantity Ranges 1
Configuring Pricing to Consume Granted Allowances Before Charging 2
Configuring Effective Dates for Pricing 3
About Defining Pricing Based on Multiple RUMs 3
About Price Overrides 4
About Fold Charge Pricing 4
Granting First Usage Balances During Rating 5
Automatically Renewing Granted Allowances 5

12 Creating Discount Offers

About Discount Offers and Discounts 1
Configuring How Discount Offers are Purchased and Applied 4
    Applying Multiple Discount Offers to an Event 5
    Configuring Mutually Exclusive Discount Offers 5
Applying Discounts Based on Quantity 6
About Discount Filters 7
About Discount Triggers 8
About Using Temporary Balances in Discounts 9
Using Functions to Calculate a Balance Impact 9
Applying Multiple Discount Versions to an Event 10
About Billing-Time Discounts 10
Configuring Effective Dates for Discounts 10
Configuring Discount Validity 11
    Specifying Discount Behavior for Partial Cycles 11
    Applying a Discount After the Discount Offer Is Inactivated or Canceled 12
    Configuring Validity Periods for Noncurrency Credit Balance Impacts 12
Discount Validity Rounding 13
Discounts and Tax Calculation 13
Discounts and General Ledger 14
About Snowball Discounts 14
    Defining How Snowball Discounts Are Distributed 14
Applying Price Tags to Discount Offers 16
Tying Discount Offers with Charge Offers 16
    Setting Discount Offers as Eligible for Tying 17
    Configuring External Applications to Manage Discount Offer Ties 17
        Tying Discount Offers at Purchase 18
        Changing the Tied Discount for a Charge Offer 18
        Handling Untied Discount Offers 19

Copyright © 2017, 2026, Oracle and/or its affiliates.

Retrieving Discounts and Purchased Offerings by Tie ID 19

13 Creating Subscription Terms

About Subscription Terms 1
About Penalty Fees 1
About Automatic Renewals 2
About Bundle Terms and Package Terms 2
About Obsoleting Subscription Terms 3
Creating Subscription Terms 3

14 Creating Bundles

About Creating Bundles 1
Configuring Bundle Settings 2
About Activating First-Usage Discount Offers 3
About Granting First-Usage Balance Impacts In First-Usage Charges 3
Associating Terms with Bundles 4
Customizing Offers In Bundles 4
About the Validity Periods of Offers in Bundles 5
Setting Proration for Offers in a Bundle 6
About Add-On Charge Offers in Bundles 7
About Purchasing the Same Offer Multiple Times 7
About Customizing Charges and Discounts in Bundles 9
Creating Dependencies for Bundles 10
Transitioning between Bundles 10
Prorating Charges During Bundle Transitions 10

15 Creating Packages and Package Lists

Creating Packages 1
Associating Terms with Packages 2
Grouping Services 3
Transitioning Between Packages 3
Prorating Charges During Package Transitions 3
Defining Generation Changes 4
Creating Package Lists 4
Specifying the Package List Segment and Type 5

16 Sharing Charges and Discounts

About Sharing Discounts and Charges 1

Copyright © 2017, 2026, Oracle and/or its affiliates.

About First-Usage Start Time for Shared Balances 2
About Discount Sharing 2
About Chargeshare Offers 2

17 Creating Configurable Recurring Offers

About Creating Configurable Recurring Offers That Can Be Changed At Purchase 1
About the Product Specification Characteristic Attributes You Can Expose 1
How Configurable Recurring Offers Work 2
Loading the Product Specification Characteristic Templates 2
Setting Up Configurable Recurring Offers at Purchase 2
Configuring External Applications to Manage Product Spec Characteristic Attributes 3
About Limitations When Using Configurable Recurring Offers 4

Part III Charging Scenarios

18 Charging Based on Date and Time

About Charging Based on Date and Time 1
About Special Day Calendars 1
Rating Events Split Across Time Periods 2

19 Charging Based on Event Attributes

About Impact Categories 1
About Zone Models 2
Alternate Zone Models 3
Displaying Zoning Information on an Invoice 4
About Value Maps 4
About Selectors 5
Impact-Based Candidate Selection 6
About Usage Scenario (USC) Selectors 7
About Mapping Services 7
Mapping Service Codes and Service Classes 8
Mapping Usage Classes 8

20 Implementing Recurring Charges

About Recurring Charges 1
About Cycle Forward Charges 1
About the First Cycle Forward Charge in a New Account 2
About Cycle Arrears Charges 2

Copyright © 2017, 2026, Oracle and/or its affiliates.

About Cycle Forward Arrears Charges 3
About Implementing Cycle Forward Arrears Charging 4
About Creating Recurring Charges in PDC 4
Specifying When Recurring Charges are Charged 4
Allowing Recurring Charges to Be Prorated 5
About Recurring Charges Events 6
About Configurable Recurring Charges 6

21 Advanced Configuration for Recurring Charges

Configuring Flexible Cycle Forward Charges 1
Charging Cycle Forward Fees Associated with Flexible Cycles 2
Applying Cycle Forward Fees in Parallel 2
About Enforcing Cycle Forward Fee Processing Prior to Billing 4
About Aggregating Service Charges to Account Items 5
Handling Skipped Billing 6
Using the pin_bill_day Script to Apply Parallel Cycle Forward Fees 6
About Limitations and Impacts of Applying Cycle Forward Fees in Parallel 7
Applying Recurring Charges Based on Charge Offer Priority 7
Customizing the Cycle Interval for Specified Charge Offers 8

22 Working with Extended Rating Attributes

About Extended Rating Attributes 1
About Sharing ERAs 1
Creating Friends and Family ERAs 1
Default ERAs 1

23 Configuring Closed User Groups

Configuring Closed User Groups 1

Part IV Managing Balances

24 Managing Balances During Charging

Tracking Balances in Balance Groups 1

25 Managing Sub-Balances

About Sub-Balances 1
About Noncurrency Sub-Balances 2

Copyright © 2017, 2026, Oracle and/or its affiliates.

How Balances in Validity-Based Sub-Balances Are Updated 3
Restricting the Validity of Noncurrency Sub-Balances That Start on First Usage 4
Configuring Custom Sub-Balances 5
Defining and Loading Custom Sub-Balances 7
Specifying the Order in Which Sub-Balances Are Consumed 8
Setting the Default Consumption Rule 10
Configuring the Default Consumption Rule in ECE 11
Deleting Expired Sub-Balances 12
Configuring Time-Stamp Rounding for Cycle Events 12
Configuring Time-Stamp Rounding for Purchase Events 13

26 Configuring Dynamic Quota

Configuring Dynamic Quota 1
Configuring Dynamic Quota Selectors 6

27 About Folds

About Folds 1
About Applying Folds Only For Account-Level Products 2

28 Applying Credit Limits to Balances

About Credit Limits 1
About Credit Thresholds and Floors 1
Handling Credit Limit Conflicts 2
Allowing Customers to Exceed Their Credit Limit 2
Configuring the Maximum Number of Subscription Retries 3

29 About Rollovers

About Rollovers 1
When Rollover Events Occur 2
About Rolling Over Noncurrency Balances during Package Transition 2
About Rolling Over Balances That Expire in Midcycle 3
Prorating Rollover Balances 4
Rollover Example 5

30 Configuring Balance Impact Rounding

About Balance Impact Rounding 1
About Rounding Rules 1
How BRM Applies Rounding 2

Copyright © 2017, 2026, Oracle and/or its affiliates.

About Rounding and A/R Actions 4
About Rounding Billed and Unbilled Items 4
About Rounding for Specific Event Types 4
About Rounding Counter Balances for Discounting 4
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 Balance Impact Rounding 7
About Configuring Rounding Rules 7
About Rounding Mode Values in the BRM API 7
Configuring Rounding Rules for Counter Balances 8
About Rounding Modes for Discount Expressions 8
Configuring to Record Rounding Differences in the G/L 9
Defining a G/L ID for Rounding Differences 9
Mapping the Rounding G/L ID to an Event 10
Configuring BRM to Record Rounding Differences 11
Rounding Examples 11
Correcting for Precision Loss When Rounding Down 12
Rounding Using Different Modes 13

Part V XML Pricing Interface for PDC

31 Importing and Exporting Pricing and Setup Components

About Using the XML Pricing Interface to Create Product Offerings 1
About the ImportExportPricing Utility 2
About Dependencies between Pricing and Setup Components 3
About the XSD Files 4
Prerequisites for Using the ImportExportPricing Utility 4
Exporting Pricing and Setup Components from PDC 5
Exporting All Pricing and Setup Components 5
Exporting a Subset of Pricing and Setup Components 5
Example 1: Exporting a Subset of Pricing Components 6
Example 2: Exporting Only Setup Components Managed in BRM 6
Example 3: Exporting Only G/L IDs 6
Example 4: Exporting All Pricing Reference Objects 6
Example 5: Exporting Only Metadata Objects 6
Example 6: Exporting Setup Components Modified After a Specific Time 6
Example 7: Exporting Pricing Components Modified by Specific Users 7
Example 8: Exporting Only Extended Attribute Template Objects 7

Copyright © 2017, 2026, Oracle and/or its affiliates.

Exporting Promoted and Failed Components 7
Exporting Only Obsoleted Components 7
About Supported Pricing and Setup Components 7
Importing Pricing and Setup Components 9
Importing from a Single XML File 10
Importing Multiple XML Files from a Directory 10
Checking the Status of an Import Operation 11
Listing Pricing or Setup Components 12
Publishing Setup Components 12
Deleting Setup Components 12
Deleting Pricing Profiles 12
Keeping Only the Latest Versions of Successfully Promoted PDC Components 12
Obsoleting PDC Components 13

32 XML Examples of Creating Pricing Components

Bundle Configuration Examples 1
Activating Offers in Bundles 1
Configuring Add-On Charge Offers In Bundles 2
Configuring Proration for Offers in Bundles 4
Purchasing the Same Offer in a Bundle Multiple Times 4
Setting Charge Offer Cycle Alignment for Reactivated Bundles 8
Transitioning Bundles 8
Charge Offer Configuration Examples 9
Aligning Recurring Charges and Product Validity to a Specific Day of the Month 9
Enforcing or Exceeding Credit Limits 11
Offering Hourly Balance Impacts 16
Consuming Granted Allowances Before Applying Usage Charges 17
Prorating Recurring Fees and Rollovers 18
Setting Full Day Proration 19
Setting Taxation for Charges 21
Using a Tax Exemption Selector 22
Sharing Products Automatically with Sharing Group Members 23
Using Date Ranges for Versioning 24
Applying Price Tags to Charges 25
Specifying Validity Period to Start from Event Occurrence Date 28
Configuring the Automatic Renewal of Allowances 28
Discount Offer Configuration Examples 30
Consuming Noncurrency Resources for Discounts in Order of Expiration 30
Sharing Discount Offers Automatically with Sharing Group Members 32
Using a Tax Selector 32
Configuring Discount Validity Rounding 34

Copyright © 2017, 2026, Oracle and/or its affiliates.

Applying Price Tags to Discounts 35
Using Instant Debit in Discount Balance Impacts 36
Marking Discount Offers as Eligible for Tying with Charge Offers in XML 37
Package Configuration Examples 38
Activating Offers in Packages 38
Setting Loan Thresholds for Packages 39
Transitioning Packages 40
Selector Configuration Examples 41
Configuring Tax Selectors 41
Configuring Tax Exemption Selectors 42
Configuring Impact-Based Candidate Selection by Using the XML Pricing Interface 44

33 PDC XML Tags

Charge Offer 1
Discount Offer 16
Chargeshare Offer 31
Bundle 43
Subscription Terms 49
Time Model 51
Charge Selector 53
Generic Selector 56
Price Selector 58
Discount Selector 60
USC Selector 62
APN Selector 66
Package 68
Package List 70
Tax Selector 71
Tax Exemption Selector 72
Price Tag Setup Component 73

Part VI Customization and Administration

34 Enabling Charging for Custom Events

Overview of Enriching Event Definitions 1
About Charging Operation Types 1
About Charging Operation Type Versions 2
About Mapping Network Attributes to Event Attributes 3
Before Enriching Event Definitions 4
Enriching Event Definitions 4

Copyright © 2017, 2026, Oracle and/or its affiliates.

Editing Event Definitions 13

35 Using the PDC Web Service

About the PDC Web Service API 1
About Using the PDC Web Service 1
Creating and Modifying Pricing Components 2
About Creating the XML Files 2
Sample Code to Call the PDC Web Service 3
CreatePricing 4
ModifyPricing 4
CreatePricingAndSubmit 4
ModifyPricingAndSubmit 4
createBusinessConfig 5
createBusinessConfigAndSubmit 5
modifyBusinessConfig 5
modifyBusinessConfigAndSubmit 5
createMetadata 6
createCustomFields 6
Exporting Components 6
Creating RetrieveInputXMLType Object Using Setter Method 7
Creating RetrieveInputXMLType Object Using XML 7
Parameters for Exporting PDC Components 7
Web Service Response 8
Processing the Response from Export/GET API 9
getPricingObjects 10
getMetadataObjects 10
getCustomFields 10
getBusinessConfigObjects 10
getBRMObjects 11

36 Working with Profiles

About Pricing Profiles 1
About Default Pricing Profiles 1
About Customizing Pricing Profiles 2
About Pricing Types 3
About the PDC Target Engine Profiles 4
BRM Online Rating Target Engine Profile 4
BRM Offline Rating Target Engine Profile 5
Subscription Engine Profile 7
ECE Rating Target Engine Profile 8

Copyright © 2017, 2026, Oracle and/or its affiliates.

ECE Subscription Engine Profile 10
Target Engine Profiles for the Default Pricing Profiles 10
About Associating Custom Pricing Profiles with Target Engine Profiles 11
About Importing Custom Pricing Profiles into PDC 11
Setting Up Custom Pricing Profiles 11
Creating Custom Pricing Profiles 11
Associating Custom Pricing Profiles with Target Engine Profiles 12
Importing Custom Pricing Profiles into PDC 13

37 Configuring Product Specification Attributes for Pricing Components

About Product Specification Attributes for Pricing Components 1
Configuring Product Specification Attributes for Pricing Components 2
Using Product Specification Attributes in Pricing Components 3
About Validating Product Specification Attributes for Pricing Components 5
Modifying Product Specification Attribute Definitions 6

38 Replicating Data Between PDC Systems

About Replicating Data 1
Replicating Setup Components Managed in BRM 1
Replicating Data Between PDC Databases 2

Part VII Pricing Utilities

39 Pricing Utilities

ImportExportPricing 1
Export Mode 2
Import Mode 7
Publish Mode 9
Keep Mode 10
Delete Mode 10
Type Mode 11
Changeset Status Mode 11
PDCWalletUtil.sh 12
PDCBRMWalletUtil.sh 14
load_pin_snowball_distribution 15
load_pin_spec_rates 16
load_pin_sub_bal_contributors 17
pin_discount_cleanup 18

Copyright © 2017, 2026, Oracle and/or its affiliates.

A New Pricing Terminology

New Pricing Terminology A-1

Copyright © 2017, 2026, Oracle and/or its affiliates.

About This Content

This guide describes how to create product offerings for Oracle Communications Billing and Revenue Management (BRM) in Pricing Design Center (PDC).

Audience

This guide is intended for pricing analysts and others involved in pricing, rating, and rerating.

Copyright © 2017, 2026, Oracle and/or its affiliates.

1

About Creating Product Offerings

Learn how to create product offerings for Oracle Communications Billing and Revenue Management (BRM) in Pricing Design Center (PDC).

Topics in this document:

- About Pricing Components
- About Pricing Setup Components
- Prerequisites for Creating Product Offerings
- How PDC Tracks and Publishes Component Changes
- Making Changes to Your Product Offerings
- About Product Offering Ownership

PDC includes sample product offerings for a broadband service and for a telco service in the *PDC_home***/apps/samples/data** directory, where *PDC_home* is the directory in which PDC is installed.

About Pricing Components

BRM allows you to configure product offerings that you sell to your customers. You configure the prices and terms for the products you sell in PDC.

In PDC, your product offerings are called *packages*. A package groups charges and discounts together. Packages can also be associated with terms that govern the agreement between you and the customer who purchases the package.

Before assembling a package, you use PDC to create the pricing components that constitute the package:

- **Charge offers**: Charge offers define the charging rules, deliverables, and taxes for using your services. For example, a charge offer could specify to:
  - Charge $35 per month for an online newspaper service
  - Grant 25 newspaper articles per month
  - Charge 10 cents per minute for mobile voice usage
  - Charge a one-time $50 installation fee
- **Discount offers**: Discount offers define the rules for applying discounts to charges. For example, a discount offer could specify to:
  - Apply a 50% discount to the first six months of a newspaper subscription
  - Provide the first 500 minutes of a service for free
  - Provide every fifth movie download for free
- **Terms**: Subscription terms specify how long the customer must commit to the service, what happens if they cancel early, and when the subscription period ends.

Copyright © 2017, 2026, Oracle and/or its affiliates.

For example, the terms for a telephony service might specify a 1-year commitment with a 1-month grace period, after which an early termination fee of $25 is applied. The subscription automatically renews after one year.

After you create charge offers, discount offers, and terms, you group them into the following pricing components:

- **Bundles**: Bundles group charge offers and discount offers together for a single service. For example, you could create:
  - A bundle for a cable service that charges $50 per month for a standard subscription and premium channels, and provides a 20% discount for the first six months of the standard subscription.
  - A bundle for an Internet service that charges $65 per month for high-speed internet and $9 per month for a modem rental.

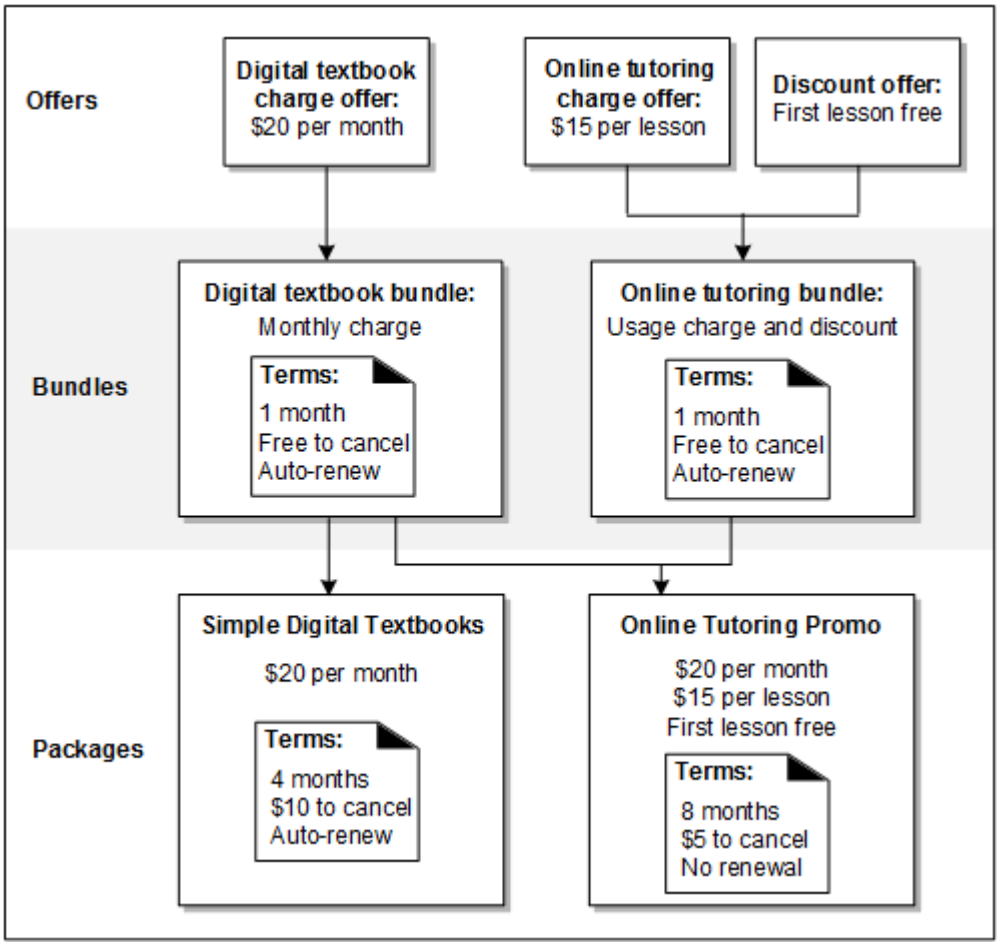
  Bundles can also be associated with terms.
- **Packages**: Packages contain one or more bundles. You use packages to offer multiple services to subscribers. For example, you can include two bundles: one for an Internet service and one for a conferencing service.

  Packages can also be associated with terms, and those terms govern any required bundles. Optional add-on bundles can be governed by separate, preset terms.
- **Package lists**: Package lists contain a group of packages offered to a single type of subscriber, such as those in a regional group.

Figure 1-1 shows how charge offers and discount offers are organized into bundles and packages with associated terms.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Figure 1-1 Grouping Offers into Bundles and Packages



About Pricing Setup Components

Before configuring your product offerings, you first configure prerequisite components defining the services you are selling, the events your customers are charged for, the tax codes to apply, and the revenue tracking information. In BRM, these prerequisite components are called *pricing setup components*. You might need users with specialized skills to configure them.

You use PDC to create these pricing setup components:

- **Services:** Services define what you base your charges on. For example, the GSM service definition enables you to charge for usage based on a customer's phone number.
- **Events**: An event is an online action, such as a phone call, an SMS message, or a data download. Data about the event is recorded in BRM. For example, when your customer makes a phone call, an event is generated that contains information about that phone call, such as the date and time the call was made. Events are also recorded for customer actions, such as purchasing a charge offer.
- **Ratable usage metrics (RUMs)**: RUMs specify how to measure events for rating, such as:
  - Duration: The length of an event in units such as seconds or minutes.
  - Occurrence: The number of events generated during a specified period.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **Service-event maps**: Service-event maps define which events can be used for a service and which RUMs can be used for each service-event combination. These maps determine which events are available when you define charge and discount offers.
- **Balance elements**: Balance elements are currency or noncurrency assets, such as U.S. dollars, minutes, or gigabytes. They can also be counters, such as tokens or points, that track the number of dollars spent or minutes talked.
- **Deliverables**: Deliverables specify when and how much revenue can be recognized from a customer's purchase of your products and services. For example, a textbook's deliverable could specify that 100% of the purchase price is recognized when the book ships from the warehouse.
- **Tax codes**: Tax codes specify the percentage of tax to apply to a charge, the jurisdiction (state or local), and how to calculate the charge.
- **Tax exemption codes**: Tax exemption codes specify the percentage by which to reduce the taxes owed for a charge.

You can also define these pricing setup components for additional charging functionality:

- **Time Models**: Time models are a set of time periods, such as 8pm to midnight and midnight to 6am. You use time models to charge different prices for a service based on the day and time.
- **Special day calendars**: Special day calendars define a set of dates, such as holidays or weekends. You add special day calendars to time models to charge different prices based on the calendar date.
- **Impact categories**: Impact categories are a set of event attributes, such as a call's origin and destination location. You use impact categories to apply different rates to events based on their attributes. For example, calls from America to Italy could be charged 10 cents per minute, and calls from America to Canada could be charged 9 cents per minute.
- **Zone models**: Zone models are a set of time zones, such as Pacific Time or Eastern Time. You use zone models to charge different prices based on your customer's time zone.
- **Value maps**: Value maps are hierarchical structures of zones and zone values. Each zone contains values that apply to a single event attribute, such as an area code, an IP address, or a broadband bit rate.
- **Selectors**: A selector is a series of rules that associates the values of event attributes, service attributes, account attributes, or custom rules to a result. Selectors can be applied to charges, discounts, items, taxes, and tax exemptions.
- **Policy specifications**: Policy specifications define a gradation of QoS based on the subscriber's service usage.
- **Extended attribute templates**: Extended attributes store additional information used by external systems, such as an enterprise product catalog. When using the XML Pricing Interface, you create extended attribute templates that configure the extended attributes that can be used for pricing components.
- **Custom analyzer rules**: A custom analyzer rule is a named expression that is used in selector rules to guide a price. For example, you can create custom rules for calls to friends and family on a subscriber's birthday, or to configure supplementary services, such as call forwarding and call blocking.
- **Price tags**: Price tags override prices for charge offers or discounts at purchase time. You configure price tag setup components to restrict values and determine which balance elements price tags can be associated with.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Prerequisites for Creating Product Offerings

Before you can start creating product offerings, complete the following tasks:

- **Set up your general ledger**. Configure how your company records financial transactions for accounting purposes. See "Setting Up Your General Ledger" in *BRM Collecting General Ledger Data*.
- **Define your tax codes, tax exemption codes, and tax suppliers**. Define the amount or percentage of taxes to apply to charges, the tax jurisdictions (state or local), and the company or company division responsible for collecting taxes. See "About Calculating Taxes" in *BRM Calculating Taxes*.
- **Define provisioning tags**. Define the custom provisioning tasks needed to implement a service on the network, such as creating a voice mailbox. You must define provisioning tags if you use charge offer or discount offer provisioning. See "Creating Provisioning Tags" in *BRM Provisioning Services*.
- **Create your pricing setup components**. Configure the prerequisite components that define the services you are selling, the events to charge your customers for, the tax codes to apply, and the revenue tracking information. See "Creating Pricing Setup Components".

How PDC Tracks and Publishes Component Changes

In PDC, you create and modify setup and pricing components in the context of a changeset. A changeset tracks the changes that you make and enables PDC to publish them to target engines.

Each user can access one or more changesets. If you do not have a changeset, PDC creates one when you log in and makes it active. When you create or modify a component, PDC adds or retains the draft in the applicable changeset.

You submit a complete changeset for transformation and publication. You can also import components into a new or existing open changeset and export components to share them with other users.

After you submit a changeset for transformation and publication, you can track its processing status. The statuses are **SUCCEEDED**, **FAILED**, or **PENDING**.

For more information, see "Checking the Status of an Import Operation".

The following topics describe how PDC creates and activates changesets, publishes changesets, and imports, exports, and shares changesets.

How PDC Creates and Activates Changesets

In PDC, you create and modify setup and pricing components in the context of a changeset. A changeset tracks the changes that you make and enables PDC to publish them to target engines.

Each user can access one or more changesets. If you do not have a changeset, PDC automatically creates one when you log in and makes it active. When you create a component, PDC adds a draft to your active changeset. If you modify a component, the updated component remains in the changeset in which it was created.

If you have multiple changesets, only one can be active. Designate the changeset you want to use as active before performing operations in the PDC application.

Copyright © 2017, 2026, Oracle and/or its affiliates.

How PDC Publishes a Changeset

You publish the components in a changeset by submitting the entire changeset. PDC processes the submitted components and sends them to the applicable target engines for transformation and publication. You cannot submit only some components in a changeset.

Submitting a changeset starts the publication process. Depending on the applicable target engines, downstream processing can continue after PDC accepts the submission.

How to Import, Export, and Share Changesets

You can import components from an XML file into a new changeset or an existing open changeset. You can also submit the imported components for publication as part of the import operation.

To share a changeset with other users, export its components to an XML file. You can optionally include components referenced by the exported components, including referenced components that are not part of the changeset.

Making Changes to Your Product Offerings

You can make changes to your product offerings at any time. For example, if you offer a new service, you can create new product offerings to charge for that service.

You can also change charges for existing charge offers, add charge offers to existing bundles, and so forth.

Deleting Charge Offers

You cannot delete a charge offer if any account owns it. To delete a charge offer, cancel it in all accounts.

Logging Changes to Product Offerings

To get notified when product offerings change, BRM can create messages in the **cm.pinlog** file when product offerings are updated. For information about **cm.pinlog** and other log files, see "Using Logs to Monitor Components" in *BRM System Administrator's Guide*.

To log product offering 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 product offerings.

> **Note**
>
> Charge offers that are used while rating events are cached by rating. The cache is refreshed when a charge offer 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** object POID that was refreshed.

Copyright © 2017, 2026, Oracle and/or its affiliates.

To log product offering 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.

About Product Offering Ownership

During account creation, the customer chooses a package. A package is a set of bundles, each of which, in turn, is a set of charge offers. Therefore, what the customer owns is not a package, but a collection of charge offers and discount offers.

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

For example, you might have two customer accounts that use the same service but have different charge offers, as shown in the following example:

Account 1

Broadband charge offer 1:

- Service: IP
- Cycle forward event: $20
- Usage events: $2

Account 2

Broadband charge offer 2:

- Service: IP
- Cycle forward event: $10
- Usage events: $1

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

When you manage a customer's account, you manage their charge offers. For example, you can change the charge offer status or customize the charge offer's pricing. When you create your product offerings, it is essential to consider how the charge offers will be managed after they have been purchased.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Part I

Creating Pricing Setup Components

This part describes how to create Oracle Communications Billing and Revenue Management (BRM) pricing setup components in Pricing Design Center (PDC).

This part contains the following chapters:

- Creating Pricing Setup Components
- Synchronizing Pricing Setup Components
- Configuring Deliverables
- Configuring Policy Specifications
- Configuring Item Type Selectors

Copyright © 2017, 2026, Oracle and/or its affiliates.

2

Creating Pricing Setup Components

Learn how to create the Oracle Communications Billing and Revenue Management (BRM) pricing setup components in Pricing Design Center (PDC).

Topics in this document:

- About Setup Components
- Configuring Setup Components
- Configuring Ratable Usage Metrics (RUMs)
- Setting Up Service and Event Definitions
- Setting Up the Service-Event Map
- Configuring Balance Elements
- Configuring Price Tags
- About Offer Profiles

About Setup Components

Setup components are the prerequisite data that you configure before creating product offerings. For example, before creating product offerings, you must configure the ratable usage metrics (RUMs) that define how to measure events.

Some setup components are defined in PDC, and some components are managed in BRM and need to be loaded into PDC.

The following setup components are defined in PDC. Any modifications to these components are also handled by PDC and are transformed into the BRM database and Oracle Communications Elastic Charging Engine (ECE).

- Balance elements
- Custom rules
- Deliverables
- Impact categories
- Item type selector
- Policy specifications
- Profile attribute specifications
- Ratable Usage Metrics (RUMs)
- Service-event maps
- Special day calendars
- Tax codes
- Tax exemption codes
- Value maps

Copyright © 2017, 2026, Oracle and/or its affiliates.

- Zone models
- Extended attribute templates
- Price tags

The following setup components are managed in the BRM server and ECE. Any modifications to these components are also handled by the BRM server and ECE and must be synchronized with PDC.

- Account definition
- Business profiles
- Event definitions
- General ledger (G/L) IDs
- Provisioning tag names
- Service definitions
- Tax suppliers

> **Note**
>
> Some of the pricing setup data is optional. You must configure the optional pricing setup data only if you plan to use the related feature or perform certain business functions. For example, you configure a zone model only if you use zoning to determine a price in your charge offer.

Configuring Setup Components

You configure the setup components in PDC by using the following applications:

- **PDC UI.** Use to create and modify setup components using a graphical user interface. See *PDC Online Help* for more information.
- **ImportExportPricing utility.** Use to import setup components defined in an XML file into PDC. See "Importing and Exporting Pricing and Setup Components" for more information.
- **SyncPDC utility.** Use to synchronize setup components defined in BRM with PDC. See "Synchronizing Pricing Setup Components" for more information.

Table 2-1 lists the setup components, the order in which you should configure the data, whether the setup component is required or optional, and the application you use to configure them.

Table 2-1 Setup Components and the Application You Use to Configure Them

Pricing Setup Data	Required/Optional	Application
RUMs	Required	PDC UI or **ImportExportPricing** utility
Service-event maps	Required	PDC UI or **ImportExportPricing** utility
Impact categories	Optional	PDC UI or **ImportExportPricing** utility
Zone models	Optional	PDC UI or **ImportExportPricing** utility
Special day calendars	Optional	PDC UI or **ImportExportPricing** utility

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 2-1 (Cont.) Setup Components and the Application You Use to Configure Them

Pricing Setup Data	Required/Optional	Application
Services	Required	Defined in BRM and pushed to PDC by the **SyncPDC** utility Modified in PDC by the **ImportExportPricing** utility
Events	Required	Defined in BRM and pushed to PDC by the **SyncPDC** utility Modified in PDC by the **ImportExportPricing** utility
Profile attributes	Required	Defined in BRM and pushed to PDC by the **SyncPDC** utility Modified in PDC by the **ImportExportPricing** utility
Account attributes	Required	Defined in BRM and pushed to PDC by the **SyncPDC** utility Modified in PDC by the **ImportExportPricing** utility
Tax codes	Optional	PDC UI or **ImportExportPricing** utility
Tax suppliers	Optional	Defined in BRM and pushed to PDC by the **SyncPDC** utility Modified in PDC by the **ImportExportPricing** utility
Tax exemption codes	Optional	PDC UI or **ImportExportPricing** utility
G/L IDs	Required for some target engines	Defined in BRM and pushed to PDC by the **SyncPDC** utility Modified in PDC by the **ImportExportPricing** utility
Deliverables	Required for deliverable-based revenue recognition	**ImportExportPricing** utility
Provisioning tag names	Optional	**SyncPDC** utility or **ImportExportPricing** utility
Balance elements	Required	PDC UI or **ImportExportPricing** utility
Value maps	Optional	**ImportExportPricing** utility
Custom analyzer rules	Optional	PDC UI or **ImportExportPricing** utility
Profile attribute specifications	Optional	**ImportExportPricing** utility
Item type selector	Required for ECE	PDC UI or **ImportExportPricing** utility
Business Profile	Required for BRM and ECE	**SyncPDC** utility
Policy Specifications	Optional	PDC UI or **ImportExportPricing** utility
Extended attribute templates	Optional	**ImportExportPricing** utility
Price tags	Optional	**ImportExportPricing** utility

Sample XML files for each setup component are in the *PDC_home*/**apps/Samples/Examples** directory, where *PDC_home* is the directory in which the PDC software is installed.

The **Examples** directory contains the following sample and ready-to-use files:

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **PDCSampleBE.xml**: Contains all currency balance elements and some frequently used noncurrency balance elements. You can load this file to create balance elements to use when configuring pricing quickly.
- **PDCSampleRum.xml**: Contains commonly used ratable usage metrics (RUMs). You can load this file to quickly create RUMs for creating a service-event map and configuring pricing.
- **OOB_ProfileSpecifications.xml**: Contains profile attribute specifications referenced in the following custom rules: **Friends&Family**, **SpecialDay**, and **ClosedUserGroup**. This XML file references the names used in the provisioning tags provided by default with the BRM installation.

  > **Note**
  >
  > You must load the **OOB_ProfileSpecifications.xml** file before loading the **OOB_CRs.xml** file.

- **OOB_CRs.xml**: Contains the custom rules: **Friends&Family**, **SpecialDay**, and **ClosedUserGroup**. You can use these custom rules in a generic selector when configuring a charge based on a subscriber's Friends & Family, Special Day, or Closed User Group profile.
- **Sample_ServiceCUG_ProfileSpecification.xml**: Contains the profile attribute specification for configuring a service-level closed user group. See "Configuring Closed User Groups" for more information.
- **Sample_ServiceCUG_CR.xml**: Contains the custom rule for configuring a service-level closed user group. See "Configuring Closed User Groups" for more information.
- **OOB_ItemTypeSelector.xml**: Contains the default billing item assignments BRM supports without any additional configuration. You must load this file if you are using ECE. See "Loading the OOB_ItemTypeSelector.XML File" for more information.
- **Sample_ItemTypeSelector.xml**: Contains the item type selector rules for defining separate bill items for international and national calls. You can use this as a template to quickly create the item type selector for tracking international calls and national calls separately. See "Configuring Item Type Selectors" for more information.
- **SampleDeliverable.xml**: Contains sample deliverables. You can use this as a template to add and modify the elements as required. See "Configuring Deliverables" for more information.
- **SamplePolicyLabel.xml**: Contains policy labels referenced in the policy specifications. You can use this as a template to add and modify the elements as required. See "Configuring Policy Specifications".
- **SamplePolicySpecification.xml**: Contains policy specifications referenced in the charge offers and discount offers used for policy-driven charging. You can use this as a template to add and modify the elements as required. See "Configuring Policy Specifications".
- **Sample_ProductSpecCharacteristicTemplate.xml**: Contains sample extended attribute configurations that can be used in charge offers, discount offers, chargeshare offers, bundles, and packages. You can use this as a template to add or modify extended attributes as required. See "Configuring Product Specification Attributes for Pricing Components".
- **Sample_Tax_Code.xml**: Contains sample tax codes for both custom and Vertex tax calculation. You can use this as a template to add or modify tax codes. See "Creating Tax Codes Using ImportExportPricing" in *BRM Calculating Taxes*.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **Sample_Tax_Exemption_Code.xml**: Contains sample tax exemptions. You can use this as a template to add or modify tax exemptions to be applied at the product level. See "Creating Product-Level Tax Exemption Codes Using ImportExportPricing" in *BRM Calculating Taxes*.
- **Sample_Price_Tag_Config.xml**: Contains sample price tag configurations. You can use this as a template to add or modify price tags to be used with charge offers and discount offers. See "Configuring Price Tags".

You can load the updated sample XML files into PDC using the **ImportExportPricing** utility. See "ImportExportPricing" for the utility's syntax and parameter descriptions.

> **Note**
>
> The **Examples** directory also contains sample pricing profile files that are provided only for reference and for use as templates for creating custom pricing profiles. You should not load the sample pricing profile files after installing PDC.

Configuring Ratable Usage Metrics (RUMs)

RUMs are ways in which you can measure events. The most common RUMs are Duration and Volume.

Before you create product offerings, you must define the RUMs available for rating. You define RUMs when you create the service-event map. For each event, you specify the RUMs that can be used for rating it.

When you define RUMs, you define the following RUM attributes:

- The unit of measurement. For example, to rate duration, you might specify seconds or minutes. To measure bytes, you might specify megabytes or kilobytes.
- How to calculate the quantity. In PDC, use the RUM expression. For example, if you specify Minute as the unit of measurement, and you specify 1 as the expression, the duration is calculated in 1-minute increments.

  You can also perform calculations on fields in the event by using the following operators:

  - Add: +
  - Subtract: -
  - Multiply: *
  - Divide: /
- How to round quantities:
  - Half Up: A fraction of .5 or greater is rounded up. For example, 1.5 is rounded to 2.
  - Half Down: A fraction of .5 or less is rounded down. For example, 1.5 is rounded to 1.
  - None
  - Round Up: Any fraction is always rounded up.
  - Round Down: Any fraction is always rounded down.
- Type:
  - Whole: Select this type if the RUM cannot be split; for example, Occurrence.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- Partial: Select this type if the RUM can be split; for example, Duration.
- Conditional: Select this type to configure charges with conditional balance impacts.

> **Note**
>
> If you are defining a RUM based on a custom BRM event attribute or a custom BRM event field, you must modify the XSD validation rules in the **PinTypes.xsd** file (*BRM*/**xsd/PinTypes.xsd**) to support this. For example, for defining the below BRM fields:
> PDC FIELD : CCBS_FLD_CCBS_GPRS
>
> BRM FIELD : CCBS_FLD_CCBS_GPRS
>
> The following validation in the XSD file should be commented or removed:
>
> ```
> <!--<xs:pattern value="[0-9]+\.*[0-9]*" />-->
> <!--<xs:pattern value="\(*PIN_FLD.+\.*PIN_FLD.*\)*.*" />-->
> <!-- <xs:pattern value="PIN_FLD_.+" /> -->
> ```
>
> Or modified, as shown below:
>
> ```
> <xs:pattern value="[0-9]+\.*[0-9]*" />
> <xs:pattern value="\(*(CCBS|PIN)_FLD.+\.*(CCBS|PIN)_FLD.*\)*.*" />
> <xs:pattern value="\(*(CCBS|PIN)_FLD.*\)*.*" />
> ```

You create RUMs by using one of the following:

- The PDC UI. For more information, see "Creating a RUM Configuration" in *PDC Online Help*.
- The **ImportExportPricing** utility. For information about creating RUMs using the utility and an XML file, see the sample RUM XML file (*PDC_home*/**apps/Samples/Examples/PDCSampleRum.xml**) and "ImportExportPricing".

Setting Up Service and Event Definitions

BRM provides a preconfigured set of service and event definitions. When you run the **SyncPDC** utility, the preconfigured service and event definitions are stored in PDC. For the list of preconfigured events, see the *BRM_home*/**sys/data/pricing/example/pin_event_map** file, where *BRM_home* is the directory in which you installed BRM.

> **Note**
>
> Before you run the **SyncPDC** utility for the first time, ensure that only those services and events that are relevant to your business are available in the **pin_event_map** file.

If you use ECE for usage rating, the preconfigured event definitions are automatically enriched with the required information for ECE when you run the **SyncPDC** utility. You can use these definitions for pricing and charging without any further enrichment. Enriching includes:

Copyright © 2017, 2026, Oracle and/or its affiliates.

- Adding user-defined charging attributes
- Defining charge operation types
- Mapping event attributes to the external network format

For example, you add specific attributes that ECE uses to apply a session-based charge for a transaction.

You can also create your own service and event definitions:

- To offer new services and events
- To add additional attributes for provisioning, networking, charging, or billing

You can define the custom services and events in Developer Center and use the **SyncPDC** utility to synchronize the definitions with PDC. If you are defining a new service or event, ensure that you do the following:

1. Create a subclass for the new service or event in Developer Center. See "Creating Service and Event Storable Classes" in *BRM Developer's Guide* for more information.
2. Create a substructure in the event subclass; for example, USAGE_INFO, and add USER_IDENTITY and CALLED_ID fields in Developer Center. See "Substructure" and "Creating Custom Fields" in *BRM Developer's Guide* for more information.
3. Map the event to the corresponding service by editing the **pin_event_map** file and then running the **load_event_map** utility.
4. Synchronize the service and event definitions with PDC by running the **SyncPDC** utility. See "Synchronizing Pricing Setup Components" for more information.

   The service and event definitions are loaded into PDC.
5. If you are using ECE for usage rating, enrich the event definitions with the information ECE requires for processing usage requests. See "Enabling Charging for Custom Events" for more information.

   The enriched event definition is published to ECE.
6. If you are using ECE for usage rating and you want to use any service or account attributes for usage rating, ensure that only those service or account attributes are loaded into the ECE cache. See "Using Service and Account Attributes with ECE for Usage Rating" for more information.

Using Service and Account Attributes with ECE for Usage Rating

If you want to use any service or account attributes for ECE usage rating, ensure only those service or account attributes are loaded into the ECE cache by doing the following:

1. Export the service or account definition into an XML file by running the following command:

   ```
   ImportExportPricing -export -metadata MetaDataObjectType -n "ObjectName"
   ```

   where:

   - *MetaDataObjectType* is the metadata object type, which can be EVENT_ATTRIBUTE_SPEC, PRODUCT_ATTRIBUTE_SPEC, PROFILE_ATTRIBUTE_SPEC, or CUSTOMER_ATTRIBUTE_SPEC. If you do not specify *MetaDataObjectType*, the utility exports all metadata components from the PDC database into the XML file.
   - *ObjectName* is the name of the service or account to export to the XML file.

Copyright © 2017, 2026, Oracle and/or its affiliates.

For example, this command would export the definition for TelcoGSM to the **productattr_export_config.xml** file:

```
ImportExportPricing -export -metadata PRODUCT_ATTRIBUTE_SPEC -n "TelcoGSM"
```

2. Open the XML file in a text editor.
3. Search for the attributes you want to use for usage rating and add the following entries for each attribute:

```
<attributeItemSupportedBy>
   <targetApplicationSpecName>Pricing</targetApplicationSpecName>
   <targetApplicationSpecName>Convergent Charging</targetApplicationSpecName>
</attributeItemSupportedBy>
```

4. Save and close the file.
5. Import the XML file into PDC by running this command:

```
ImportExportPricing -import -metadata FileOrDirectoryName -ow
```

For example:

```
ImportExportPricing -import -metadata productattr_export_config.xml -ow
```

The utility loads the service or account definition into PDC.

ECE Pricing Updater publishes only the service or account attributes you want to use for usage rating to ECE. Only those attributes are loaded into the ECE cache.

Setting Up the Service-Event Map

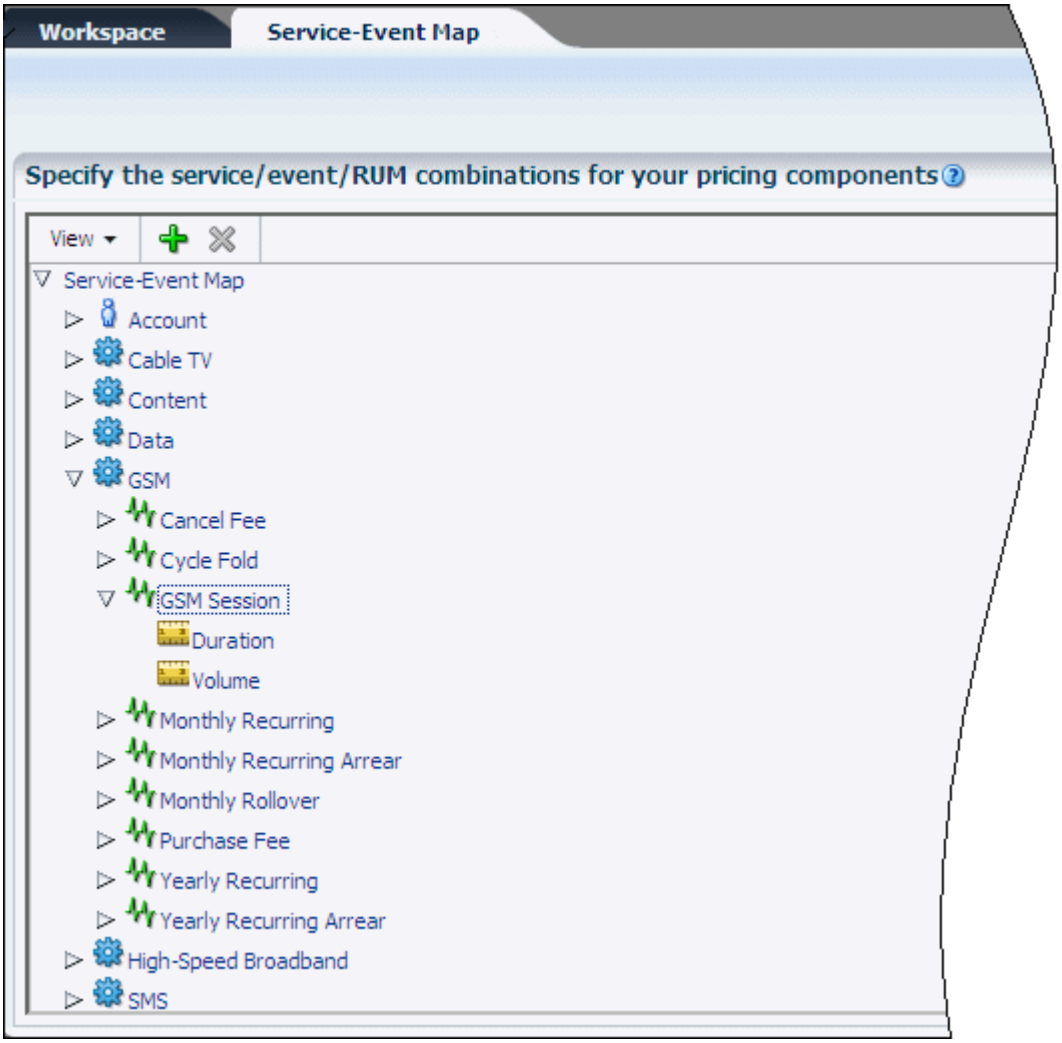
Before you can create charge offers, you must set up a service-event map. The service-event map lists all the services to which charge offers can apply. For each service, the map specifies which events can have charges configured for that service. The map also specifies the RUMs to use for rating each event combination.

For events that do not apply to a specific service, you map those events to **Account** in the service-event map.

For example, in Figure 2-1, the events Cancel Fee, Cycle Fold, GSM Session, Monthly Recurring, and Monthly Recurring Arrear are mapped to the GSM service. The RUMs, Duration and Volume, are mapped to the GSM Session event.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Figure 2-1 Service-Event Map in PDC



You can create service-event maps using the PDC UI or the **ImportExportPricing** utility.

> **Note**
>
> If you encounter a server exception when importing a Service-Event Map XML file in the PDC UI, use the **ImportExportPricing** utility to create the service-event map.

Configuring Balance Elements

Before you can create pricing components, you must use PDC to create balance elements. A balance element represents a:

- Currency, such as US dollars or euros.
- Noncurrency asset, such as minutes or gigabytes.
- Counter, such as loyalty points or tokens. Counters are noncurrency assets that allow you to track balances, such as the amount spent or megabytes downloaded by a customer.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- Temporary balance, such as minutes or gigabytes. Temporary balances are noncurrency assets that retain their balances while processing the discounts for one event. After all discounts for an event are completed, temporary balance elements are reset to 0.

When you configure pricing in a charge or a discount, you specify the balance elements that are debited or credited. For example, a charge of 1 dollar per minute for a phone call debits the USD balance element.

When you create balance elements, you define the following:

- The balance element name, such as Canadian Dollar or Free Minutes.
- The reference code for the balance element.

  For currency balance elements, the default reference code is used. For noncurrency balance elements, you must enter a unique reference code.
- The balance element ID, such as 124. Currency balance element IDs are defined in an ISO standard.
- The abbreviation or symbol used for display, such as $, Can$, or Minute.
- For noncurrency balance elements, whether used as a counter or for temporarily tracking balances.
- Whether the balance element can be used in a fold charge.
- Whether to notify customers when their balance is about to expire.
- Whether to notify customers after their balance has expired.
- The order in which to consume sub-balances. For example, if a customer's balance contains minutes with different validity periods, you specify which minutes to use first according to the starting or ending validity date. See "Specifying the Order in Which Sub-Balances Are Consumed".
- The rounding rules. See "Adding Rounding Rules to Balance Elements".

You create balance elements by using one of the following:

- The PDC UI. For more information, see "Creating Balance Elements" in *PDC Online Help*.
- The **ImportExportPricing** utility. For information about creating balance elements using the utility and an XML file, see the sample balance element XML file (*PDC_home***/apps/Samples/Examples/PDCSampleBE.xml**) and "ImportExportPricing".

For information about how to add multiple currency balance elements for subscriber currencies under a single pricing tier, see "Enabling Single Currency Graph" for more information.

Adding Rounding Rules to Balance Elements

Rounding rules allow you to control the precision of charge calculations. PDC provides a set of default rounding rules, but you can add new rounding rules for specific events.

When adding a rounding rule, you define the following:

- The processing stage in which to apply the rounding rule:
  - Accounts receivable (A/R)
  - Charging
  - Discounting
  - Taxation
- The event category that the rounding rule applies to:

Copyright © 2017, 2026, Oracle and/or its affiliates.

  - Billing-time discount
  - One-time
  - Recurring
  - Fold
  - Rollover
  - Usage
  - Remittance
- The event that the rounding rule applies to. The applicable events are based on the event category. For example, Basic Usage is an option for the Usage event category.
- If the selected event has child events, whether to use the rounding rule for both parent and child events.
- How to round additional numbers beyond the precision value:
  - Up: For example, 10.151 rounds to 10.16.
  - Down: For example, 10.159 rounds to 10.15.
  - Down Alt: Performs the same rounding as Down, except that it compensates for possible loss of precision by first rounding with a mode of Nearest using a scale two digits greater than the scale you configure. See "About Rounding Modes That Correct for Loss of Precision".
  - Nearest: The last significant digit remains the same if the additional digit is 0 through 4. The last significant digit is rounded up if the additional digit is 5 through 9. For example, 10.144 rounds to 10.14, and 10.145 rounds to 10.15.
  - Even: If the additional digit is 0 through 4, the last significant digit remains the same. If the additional digit is 6 through 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, 10.155 rounds to 10.16, and 10.165 rounds to 10.16.
  - Floor: 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.
  - Floor Alt: Performs the same rounding as Floor, except that it compensates for possible loss of precision by first rounding with a mode of Nearest using a scale that is two digits greater than the scale you configure. See "About Rounding Modes That Correct for Loss of Precision".
- The rounding precision. For example, to round US dollars to cents, specify a precision value of two places after the decimal point.
- The minimum and maximum tolerance values and the tolerance percentage that determines how the rounding rule is applied to the balance element. For example, you can specify a tolerance of 4 digits after the decimal point.

For more information about rounding balance elements, see "Configuring Balance Impact Rounding".

Enabling Single Currency Graph

To enable a single currency graph:

Copyright © 2017, 2026, Oracle and/or its affiliates.

> **Note**
>
> A single currency graph can be used only with ECE.

1. Go to the *PDC_home*/**apps/bin** directory.
2. Export the pricing data by running the **ImportExportPricing** utility.

   For example, to export the data to a file named **chargeoffer_export_pricing.xml**, run the following command:

   ```
   ./ImportExportPricing -export chargeoffer -pricing
   ```

3. In the **chargeoffer_export_pricing.xml** file, search for the **balanceElementNumCode** attribute. By default, only one currency code is available. You can add more currency codes.

   For example:

```
-<subscriberCurrency>
  <currencyCode>USD</currencyCode>
  <currencyCode>EUR</currencyCode>
  -<applicableRum>
      <applicableRumName>Duration</applicableRumName>
      <minQuantity>1.0</minQuantity>
      <minQuantityUnit>NONE</minQuantityUnit>
      <incrementQuantity>1.0</incrementQuantity>
      <incrementQuantityUnit>NONE</incrementQuantityUnit>
      <roundingMode>NEAREST</roundingMode>
      -<crpRelDateRange>
         -<absoluteDateRange>
             <startDate>0</startDate>
             <endDate>inf</endDate>
         </absoluteDateRange>
        -<crpCompositePopModel>
             <name>Pricing</name>
             -<usageChargePopModel>
               -<priceTier>
                   <distributionMethod>FROM_BAL_IMPACT</distributionMethod>
                   -<tierBasis>
                      <rumTierExpression/>
                    </tierBasis>
                    <enforceCreditLimit>false</enforceCreditLimit>
                    <rumName>Duration</rumName>
                    <currencyCode>USD</currencyCode>
                   -<priceTierValidityPeriod>
                       <lowerBound>0</lowerBound>
                       <validFrom>0</validFrom>
               -<priceTierRange>
                  <upperBound>NO_MAX</upperBound>
                 -<scaledCharge>
                    <price>0.0</price>
                    <unitOfMeasure>NONE</unitOfMeasure>
                    <balanceElementNumCode>840</balanceElementNumCode>
                    <discountable>true</discountable>
                    <priceType>CONSUMPTION</priceType>
                    <incrementStep>1.0</incrementStep>
                    <incrementRounding>NONE</incrementRounding>
                  </scaledCharge>
               </priceTierRange>
    </priceTierValidityPeriod>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
<applicableQuantity>ORIGINAL</applicableQuantity>
</priceTier>
</usageChargePopModel>
</crpCompositePopModel>
</crpRelDateRange>
</applicableRum>
</subscriberCurrency>
```

4. Save the XML file as **import_pricing.xml** file.
5. Import the **import_pricing.xml** file using the **ImportExportPricing** utility.

   For example:

   ```
   ./ImportExportPricing -import -pricing import_pricing.xml -ow
   ```

6. Add, edit, or remove balance impacts for all the supported subscriber currencies.

Configuring Price Tags

Price tags are components that let you apply special prices to charge offers or discounts according to your specified rules.

You configure price tags as setup components, add them to balance impacts or quantity ranges when creating charge offers and discount offers, and set their values at purchase time.

The rules configured in the setup components 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 charge shares or temporary balance elements in discount offers.

When you configure price tag setup components, 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 balance elements you add the price tag to. For example, Free Minutes, US Dollars, or any balance element.
- The unit of measure 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 configure price tag setup components by using one of the following:

- The PDC UI. See "Configuring Price Tags" in *PDC Online Help*.
- The **ImportExportPricing** utility. For information about creating price tags using the utility and an XML file, see the sample price tag XML file (*PDC_home***/apps/Samples/Examples/Sample_Price_Tag_Config.xml**), and ImportExportPricing".

Price tag configuration information is stored in BRM in the **/config/price_tags** storable class.

After configuring price tag setup components, you can update or delete them under the following conditions:

- If the price tag is not yet associated with any charge offers or discount offers, you can update any of the price tag's values or obsolete it.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- If the price tag is associated with charge offers or discount offers, you can update the description, rule, and value fields. You cannot update any other fields; you must create a new price tag for the new scenario. You cannot obsolete a price tag associated with a charge offer or discount offer.

You add price tags to balance impacts or quantity ranges using one of the following:

- The PDC UI. See "Adding Price Tags for One-Time and Recurring Charges", "Adding Price Tags for Usage Charges", and "Adding Price Tags for Discounts" in *PDC Online Help*.
- The **ImportExportPricing** utility. See "Applying Price Tags to Charges" and "Applying Price Tags to Discounts".

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 example, you can have an offer profile called **Platinum** for a data service and define its balance element 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.

Copyright © 2017, 2026, Oracle and/or its affiliates.

3

Synchronizing Pricing Setup Components

Learn how to synchronize Oracle Communications Billing and Revenue Management (BRM) setup components, such as services and events, with Pricing Design Center (PDC) by using the **SyncPDC** utility.

Topics in this document:

- About Synchronizing Setup Components
- About the SyncPDC Utility
- Changing Display Names

About Synchronizing Setup Components

When configuring a new PDC system, you use the **SyncPDC** utility to synchronize the setup components defined in the billing and rating system with PDC. See "Configuring Setup Components" for more information.

> **Note**
>
> Ensure that **/event/realtimeDiscount** does not exist in BRM during synchronization.

SyncPDC synchronizes the following setup components:

- Account definitions
- Business profiles
- Event definitions
- General ledger (G/L) IDs
- Profile definitions
- Provisioning tags
- Service definitions
- Tax suppliers

After these setup components are synchronized with PDC, you perform any modifications in the system in which they were initially defined and then resynchronized.

SyncPDC synchronizes only the changes from the previous synchronization. It generates reports for the components that it synchronizes. See "Generating Synchronization Reports" for more information.

Copyright © 2017, 2026, Oracle and/or its affiliates.

> **Note**
>
> After synchronizing these setup components using SyncPDC, you cannot delete them from PDC or BRM.

When synchronizing setup components, **SyncPDC** creates display names in PDC for the corresponding data objects and their fields. For example, it creates the **GsmTelephony** PDC name for the **/service/gsm/telephony** service name in BRM. You can change the PDC display name in PDC. For example, you can change **EventTelcoGsmVoice** to **Voice Call**. If you later run **SyncPDC** to resynchronize this event, it retains the name change. See "Changing Display Names" for more information about updating PDC display names.

If you delete a setup component (or any of its fields) from the billing or rating system, you must manually remove the corresponding PDC component (or its fields) from the PDC database. See "Deleting Setup Components" for more information.

About the SyncPDC Utility

The **SyncPDC** utility runs as a server process in the background, continuously checking for data to synchronize from BRM or the rating system with PDC. You can schedule it to run immediately, at a specified time, or regularly at a specified time. At any given time, you can stop the synchronization by stopping **SyncPDC**. See "Stopping SyncPDC" for more information.

Prerequisites for Running SyncPDC

Before you run **SyncPDC**, do the following:

- Start the transformation engines for the rating systems. See "Starting the Transformation Engines" in *BRM System Administrator's Guide*.
- Include the *PDC_home***/apps/bin** directory in your PATH environment variable, where *PDC_home* is the directory in which you installed the PDC software.
- Configure the *BRM_Integration_Pack_Home***/apps/syncpdc/SyncPDCConfiguration.xml** file. See Table 3-1 for more information.

To run **SyncPDC**, you must have **Pricing Design Admin** role privileges.

About Configuring SyncPDC

SyncPDC uses the **SyncPDCConfiguration.xml** file, which contains configuration settings for **SyncPDC** as well as connection information for Oracle WebLogic Server Administration, the PDC server, and the cross-reference database. This file is generated during PDC installation and is located in the *BRM_Integration_Pack_Home***/apps/syncpdc** directory. By default, this file contains the values that you provide during BRM Integration Pack installation. You can change the default values by updating this file.

To update **SyncPDCConfiguration.xml**:

1. Open the *BRM_Integration_Pack_Home***/apps/syncpdc/SyncPDCConfiguration.xml** file in a text editor or an XML editor.
2. Edit the file. See Table 3-1 for more information.
3. Save and close the file.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 3-1 lists the elements in **SyncPDCConfiguration.xml**, the usage of each element, and a description of how to specify each element based on the default version of the file.

Table 3-1 Elements in the SyncPDCConfiguration.xml File

Element	Syntax	Description
Configuration	`<Configuration>`	The root element of **SyncPDCConfiguration.xml**.
walletConfiguration	`<walletConfiguration>` `  <pdcWalletLocation>`*`BRMIntegrationPackWallet`*`</pdcWalletLocation>` `</walletConfiguration>`	Specifies the Oracle wallet that stores the sensitive information, such as database passwords, where *BRMIntegrationPackWallet* is the complete path to the Oracle wallet in BRM Integration Pack.
xrefDatabase	`<xrefDatabase>` `  <connectionInfo>` `  <login>`*`CrossRefUserName`*`</login>` `  <hostName>`*`CrossRefHostName`*`</hostName>` `  <port>`*`CrossRefPort`*`</port>` `  <serviceName>`*`CrossRefServiceName`*`</serviceName>` `  </connectionInfo>` `</xrefDatabase>`	Contains details about the cross-reference database, where: • *CrossRefUserName* specifies the cross-reference database user name • *CrossRefHostName* specifies the IP address or the host name of the computer on which the cross-reference database is configured • *CrossRefPort* specifies the port number assigned to the cross-reference database • *CrossRefServiceName* specifies the name of the cross-reference database service
pricingServer	`<pricingServer>` `  <connectionInfo>` `  <hostName>`*`PricingServerHostName`*`</hostName>` `  <port>`*`PricingServerPort`*`</port>` `  <adminUser>`*`AdminUserName`*`</adminUser>` `  <pdcUser>`*`PDCUser`*`</pdcUser>` `  </connectionInfo>` `</pricingServer>`	Contains details about the PDC server, where: • *PricingServerHostName* specifies the IP address or the host name of the computer on which PDC is deployed • *PricingServerPort* specifies the port number of the domain on which PDC is deployed • *AdminUserName* specifies the user name of the PDC server administrator • *PDCUser* specifies the PDC user name
transformationHome	`<transformationHome>`*`TransformationHome`*`</transformationHome>`	Specifies the path to the directory that stores the transformation process IDs, where *TransformationHome* is the complete path to the directory.
syncPDCLogFileLocation	`<syncPDCLogFileLocation>`*`SyncPDCLogFile`*`</syncPDCLogFileLocation>`	Specifies the path to the directory that stores **SyncPDC** log files, where *SyncPDCLogFile* is the complete path and the name of the log file.
brand	`<brand>`*`BrandOption`*`</brand>`	Specifies whether PDC supports branding, where *BrandOption* is **enabled** or **disabled**.
pdcSSL	`<pdcSSL>`*`SSLOption`*`</pdcSSL>`	Specifies whether the PDC server supports SSL, where *SSLOption* is: • **enabled** to specify that PDC supports SSL. If SSL is enabled, **SyncPDC** uses the **t3s://***Host***:***Port* URL to access PDC • **disabled** to specify that PDC does not support SSL. If SSL is disabled, **SyncPDC** uses the **t3://***Host***:***Port* URL to access PDC

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 3-1 (Cont.) Elements in the SyncPDCConfiguration.xml File

Element	Syntax	Description
retryInfo	`<retryInfo>` `  <retryInterval>`*`RetryInterval`*`</retryInterval>` `  <maxRetries>`*`MaxRetries`*`</maxRetries>` `</retryInfo>`	Contains the information about when and how many times **SyncPDC** should retry the synchronization process if it fails initially, where: • *RetryInterval* specifies the interval in seconds after which **SyncPDC** again tries to synchronize the data that was not previously synchronized • *MaxRetries* specifies the maximum number of retries
ECESync	`<ECESync>`*`ECESyncOption`*`</ECESync>`	Specifies to synchronize BRM data with PDC, where *ECESyncOption* is: • **true** to synchronize BRM data with PDC to use Oracle Communications Elastic Charging Engine (ECE) for usage rating • **false** to synchronize BRM data with PDC to use real-time and batch rating engines for usage rating
pdcXML	`<pdcXML>`*`PDCData`*`</pdcXML>`	Specifies the path to the directory where **SyncPDC** creates the XML files containing the extracted data in PDC format, where *PDCData* is the complete path to the directory.
reportFileLocation	`<reportFileLocation>`*`ReportLocation`*`</reportFileLocation>`	Specifies the path to the directory where **SyncPDC** stores the report of the synchronization process, where *ReportLocation* is the complete path to the directory.
brmExtractedXML	`<brmExtractedXML>`*`BRMData`*`</brmExtractedXML>`	Specifies the path to the directory where **SyncPDC** creates the XML files containing the data extracted from the BRM database, where *BRMData* is the complete path to the directory.
pdcExtractedXML	`<pdcExtractedXML>`*`PDCData`*`</pdcExtractedXML>`	Specifies the path to the directory where **SyncPDC** creates the XML files containing the data extracted from the PDC database, where *PDCData* is the complete path to the directory.
archiveFileLocation	`<archiveFileLocation>`*`ArchiveFile`*`</archiveFileLocation>`	Specifies the path to the directory where a successfully processed file is archived, where *ArchiveFile* is the complete path to the directory. No archive file is created if there is no change from the previous synchronization.
skipBREMigration	`<skipBREMigration>`*`SkipOption`*`</skipBREMigration>`	Specifies whether to skip synchronization of pipeline configuration data, where *SkipOption* is either **true** or **false**. This element is ignored when ECE is enabled in your PDC system, where the **ECESync** option is set to **true**. The default value is **false** when you use PDC with Realtime Rating and Batch Rating Engines.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 3-1 (Cont.) Elements in the SyncPDCConfiguration.xml File

Element	Syntax	Description
fieldSelection	`<fieldSelection>` `<targetEngine>`*`TargetRatingEngine`*`</targetEngine>` `<eventFields>` `<eventName>`*`EventName`*`</eventName>` `<fullyQualifiedName>`*`EventFieldName`*`</fullyQualifiedName>` `</eventFields>` `</fieldSelection>`	Contains the information about the target rating engine and the BRM event fields provided as input to the target rating engine for usage rating. **<targetEngine>** element specifies the target rating engine used for usage rating, where: *TargetRatingEngine* is one of the following: • **Convergent Charging**. Specifies that ECE is used for usage rating. • **Realtime Charging**. Specifies that the real-time rating engine is used for usage rating. • **Batch Charging**. Specifies that the batch rating engine is used for usage rating. **<eventFields>** element specifies the fields that are provided as input to *TargetRatingEngine*, where: • *EventName* specifies the class name of the BRM event; for example,/event. • *EventFieldName* specifies the fully qualified name of the BRM event field that is provided as input to *TargetRatingEngine*; for example, PIN_FLD_NAME. **Note:** Each **<fieldSelection>** element can have multiple **<eventFields>** elements but only one **<targetEngine>** element. To support multiple target rating engines, add the **<fieldSelection>** element for each target rating engine. Similarly, each **<eventFields>** element can have multiple **<fullyQualifiedName>** elements but only one **<eventName>** element. To rate multiple events, add the **<eventFields>** element for each event.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 3-1 (Cont.) Elements in the SyncPDCConfiguration.xml File

Element	Syntax	Description
syncPDCBREConfig	<syncPDCBREConfig> <containerDesc> <param> <paramname>*ParamName*</paramname> <paramvalue>*ParamValue*</paramvalue> </param> </containerDesc> <eventExtension> <param> <paramname>*BRMBatchRatingEventName*</paramname> <paramvalue>*EventExtensionBlockName*</paramvalue> </param> </eventExtension> <serviceExtension> <param> <paramname>*BRMServiceName*</paramname> <paramvalue>*ServiceExtensionBlockName*</paramvalue> </param> </serviceExtension> </syncPDCBREConfig>	Contains the information about EDR container description data and delayed events and services for batch rating engine. The **<containerDesc>** element specifies the EDR container description data that is used by the batch rating engine, where: • *ParamName* specifies the field name that defines the EDR container description name • *ParamValue* specifies the value of *ParamName*. The EDR container description data that matched *ParamValue* is used as the base to synchronize EDR fields for events and services The **<eventExtension>** element specifies the mapping of a delayed event to the extension block name used in the batch rating engine, where: • *BRMBatchRatingEventName* specifies the delayed event name • *EventExtensionBlockName* specifies the extension block name to which the delayed event is mapped Add **<param>** elements to the **<eventExtension>** elements for each *BRMBatchRatingEventName* and *EventExtensionBlockName* to include. The **<serviceExtension>** element specifies the mapping of a BRM service to the extension block name used in the batch rating engine, where: • *BRMServiceName* specifies the BRM service name • *ServiceExtensionBlockName* specifies the extension block name to which the BRM service is mapped Add **<param>** elements to the **<serviceExtension>** elements for each *BRMServiceName* and *ServiceExtensionBlockName* to include. For more information about the EDR container description, see the BRM documentation.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 3-1 (Cont.) Elements in the SyncPDCConfiguration.xml File

Element	Syntax	Description
scheduling	`<scheduling>` `  <runOnStartup>`*`StartupOption`*`</runOnStartup>` `  <startAt>`*`StartTime`*`</startAt>` `  <interval>`*`Interval`*`</interval>` `</scheduling>`	Contains the scheduling configurations for **SyncPDC**. If the **<scheduling>** element is not present or its child elements are not defined correctly, **SyncPDC** runs immediately only once. *StartupOption* specifies whether **SyncPDC** should run immediately: • **true** specifies that **SyncPDC** runs immediately in addition to the scheduled time • **false** specifies that **SyncPDC** does not run immediately but at the scheduled time *StartTime* specifies the time in *HH***:***MM* format when you want **SyncPDC** to start, where: • *HH* specifies hours between 00 and 23 • *MM* specifies minutes between 00 and 59 *Interval* specifies the frequency in *N***:***A* format at which you want to run **SyncPDC**, where: • *N* is an integer • *A* is **D** for days, **H** for hours, or **M** for minutes For example, to run **SyncPDC** every two days, enter: `<interval>2:D</interval>` **Note: 24:H** is not same as **1:D** due to daylight saving time.

Running SyncPDC

To run **SyncPDC**:

1. Ensure that transformation engines are running. See "Starting the Transformation Engines" in *BRM System Administrator's Guide*.
2. Go to the *BRM_Integration_Pack_Home***/apps/syncpdc** directory.
3. Enter the following command:

   `startSyncPDC`

   The **Enter Wallet Password** prompt appears.
4. Enter the BRM Integration Pack wallet password.

   A series of messages appears on the command prompt that indicate the synchronization status.

   For example:

```
Clean up work item SYNC_EVENT...
Work item SYNC_EVENT started (item 1 of 8).
  Processing EXTRACT work action...
  Work action EXTRACT completed.
  Processing ANALYZE work action...
  Work action ANALYZE completed.
  Processing TRANSFORM work action...
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
  Work action TRANSFORM completed.
Processing LOAD work action...
  Work action LOAD completed.
Work item SYNC_EVENT completed.
```

After the synchronization is complete, check the synchronization report available in the directory defined in the **<reportFileLocation>** element of the **SyncPDCConfiguration.xml** file.

Scheduling Synchronization

You can schedule the data synchronization from BRM or rating systems with PDC by:

- Running SyncPDC Immediately
- Running SyncPDC at a Scheduled Time
- Running SyncPDC Immediately and at a Scheduled Time
- Running SyncPDC at Recurring Scheduled Time

Running SyncPDC Immediately

To run **SyncPDC** immediately instead of waiting for the next scheduled run time:

1. Open the *BRM_Integration_Pack_Home***/apps/syncpdc/SyncPDCConfiguration.xml** file in a text editor or an XML editor.
2. Set the **<runOnStartup>** element to **true**:

   ```
   <runOnStartup>true</runOnStartup>
   ```

3. Save and close the file.
4. (Optional) If **SyncPDC** is already running in the background, stop it.

   See "Stopping SyncPDC" for more information.
5. Run **SyncPDC**. See "Running SyncPDC" for more information.

Running SyncPDC at a Scheduled Time

To run **SyncPDC** at a schedule time:

1. Open the *BRM_Integration_Pack_Home***/apps/syncpdc/SyncPDCConfiguration.xml** file in a text editor or an XML editor.
2. Set the **<runOnStartup>** element to **false**:

   ```
   <runOnStartup>false</runOnStartup>
   ```

3. Set the **<startAt>** element to the start time:

   <startAt>*HH:MM***</startAt>**

   where:

   - *HH* specifies hours between 00 and 23
   - *MM* specifies minutes between 00 and 59

   For example, to run **SyncPDC** at 11:50 pm, enter:

   ```
   <startAt>23:50</startAt>
   ```

4. Save and close the file.

Copyright © 2017, 2026, Oracle and/or its affiliates.

5. Run **SyncPDC**.

   See "Running SyncPDC" for more information.

Running SyncPDC Immediately and at a Scheduled Time

To run **SyncPDC** immediately and at a schedule time:

1. Open the *BRM_Integration_Pack_Home***/apps/syncpdc/SyncPDCConfiguration.xml** file in a text editor or an XML editor.
2. Set the **<runOnStartup>** element to **true**:

   ```
   <runOnStartup>true</runOnStartup>
   ```

3. Set the **<startAt>** element to the start time:

   ```
   <startAt>HH:MM</startAt>
   ```

   where:

   - *HH* specifies hours between 00 and 23
   - *MM* specifies minutes between 00 and 59

   For example, to run **SyncPDC** immediately and at 11:50 pm, enter:

   ```
   <runOnStartup>true</runOnStartup>
   <startAt>23:50</startAt>
   ```

4. Save and close the file.
5. Run **SyncPDC**.

   See "Running SyncPDC" for more information.

Running SyncPDC at Recurring Scheduled Time

To run **SyncPDC** at recurring scheduled time:

1. Open the *BRM_Integration_Pack_Home***/apps/syncpdc/SyncPDCConfiguration.xml** file in a text editor or an XML editor.
2. Set the **<runOnStartup>** element to **false**:

   ```
   <runOnStartup>false</runOnStartup>
   ```

3. Set the **<startAt>** element to the start time:

   ```
   <startAt>HH:MM</startAt>
   ```

   where:

   - *HH* specifies hours between 00 and 23
   - *MM* specifies minutes between 00 and 59

4. Specify the **<interval>** element as follows:

   ```
   <interval>N:A</interval>
   ```

   where:

   - *N* is an integer
   - *A* is **D** for days, **H** for hours, or **M** for minutes

   For example, to run **SyncPDC** at 11:50 pm every day, enter:

Copyright © 2017, 2026, Oracle and/or its affiliates.

```
<runOnStartup>false</runOnStartup>
<startAt>23:50</startAt>
<interval>1:D</interval>
```

5. Save and close the file.
6. Run **SyncPDC**.

   See "Running SyncPDC" for more information.

Synchronizing BRM Data with PDC to Use ECE for Usage Rating

To synchronize BRM data with PDC to use ECE for usage rating:

1. Open the *BRM_Integration_Pack_Home***/apps/syncpdc/SyncPDCConfiguration.xml** file in a text editor or an XML editor.
2. Set the **<ECESync>** element to **true**:

   ```
   <ECESync>true</ECESync>
   ```

3. Save and close the file.
4. Run **SyncPDC**.

   See "Running SyncPDC" for more information.

> **Note**
>
> While synchronizing multiple events for the same service in the Pipeline Manager, the **IFW_REF_MAP** table may not get synchronized to PDC. This happens if there are inconsistencies between the primary key constraint of the **IFW_REF_MAP** table and **LoadIfwConfig.xsd** configuration file. **SyncPDC** throws an error message indicating that the event is not configured as a rating event in BRM.
>
> To include multiple event mappings for the same service, you modify the **IFW_REF_MAP** table's primary key constraint to include the **REF_PARAM** column with **ID** and **REF_OBJ**.
>
> Modify the **LoadIfwConfig.xsd** file to ensure the **IFW_REF_MAP_PrimaryKey_1** constraint in the **LoadIfwConfig.xsd** file is the same as the primary key constraint in the **IFW_REF_MAP** table. To reflect these changes:
>
> 1. Go to the *Pipeline_home*/**tools/XmlLoader** directory.
> 2. Create a copy of the **LoadIfwConfig.xsd** file.
> 3. Modify the **LoadIfwConfig.xsd** file to add the **REF_PARAM** element:
>
>    ```
>    <xs:element name="IFW_REF_MAP" type="TableType_IFW_REF_MAP"
>    minOccurs="0" maxOccurs="unbounded">
>    <xs:key name="IFW_REF_MAP_PrimaryKey_1">
>    <xs:selector xpath="IFW_REF_MAP"/>
>    <xs:field xpath="@ID"/>
>    <xs:field xpath="@REF_OBJ"/>
>    <xs:field xpath="@REF_PARAM"/>
>    </xs:key>
>    </xs:element>
>    ```
>
> 4. Run the SyncPDC utility. See "Running SyncPDC" for more information.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Stopping SyncPDC

You can stop **SyncPDC** at any time. If you stop **SyncPDC** while it is synchronizing data, it finishes the synchronization before stopping.

To stop **SyncPDC**, enter the following command:

```
stopSyncPDC
```

A series of messages appears, indicating progress.

For example:

```
Previous process id :4381
Killing child processes ...
Killing process ...
Cleaning temporary files ...
[1]  + Killed
```

Generating Synchronization Reports

SyncPDC generates reports for the components that it synchronizes. If there are no changes from the previous synchronization, **SyncPDC** does not create a report.

You specify the location of the **SyncPDC** report in the **<reportFileLocation>** element of the **SyncPDCConfiguration.xml** file.

When synchronizing data from BRM with PDC, the **SyncPDC** report lists the object type for each setup component that is synchronized. It lists the PDC display name and the corresponding BRM or rating system name for each of the setup component objects. It also provides the total number of object types for each setup component that are synchronized.

Figure 3-1 shows a sample **SyncPDC** report for BRM data.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Figure 3-1 Sample SyncPDC Report for BRM Data

SyncPDC Report. Generated on 20120730_235025_832

Report Type: SyncPDC, Object Type: Event

PDC Internal Id	PDC Event Name	BRM Event Name
d672f756-ed1c-4be8-93dd-ea4f787ea809	Event	/event
9a34af04-9f86-4c5f-8b99-b884c6c47490	EventActivityContent	/event/activity/content
2c65f382-ad3a-4920-b30d-63785e3ef954	EventActivitySettlement	/event/activity/settlement
b7338796-3c4c-4564-88ca-15cc5e289ecd	EventActivityTelco	/event/activity/telco
51a87729-e4ab-466f-80de-9d37f7890aa8	EventBillingCycleDiscount	/event/billing/cycle/discount
c0576f18-54cc-4f8a-9ed6-754ca44dbead	EventBillingCycleDiscountMostcalled	/event/billing/cycle/discount/mostcalled
b05c6917-305d-4b43-bb3a-b9f6faa6b9c1	EventBillingCycleFold	/event/billing/cycle/fold
b260e6d5-601e-442b-9f96-f56c58e11df7	EventBillingCycleRolloverMonthly	/event/billing/cycle/rollover/monthly
51b0ae22-7101-4b47-af55-1a122ea48419	EventBillingFeeFailed_payment	/event/billing/fee/failed_payment
c37e96ed-0202-47ee-a45a-043dc8699adb	EventBillingIncentive	/event/billing/incentive
f13ad160-3093-4a60-9574-a3957952ea86	EventBillingProductFeeCancel	/event/billing/product/fee/cancel
3bad8520-2c95-4b1c-a22b-78c87b3e0cfb	EventBillingProductFeeCycleCycle_arrear	/event/billing/product/fee/cycle/cycle_arrear
927f9fca-f02a-4e16-98ce-ea8c12d193ea	EventBillingProductFeeCycleCycle_forward_annual	/event/billing/product/fee/cycle/cycle_forward_annual

Total number of BRM event objects synchronized: 13

When synchronizing data from ECE with PDC, the **SyncPDC** report provides the status of loading the XML files containing ECE data.

Figure 3-2 shows a sample **SyncPDC** report for ECE data.

Figure 3-2 Sample SyncPDC Report for ECE Data

Start Load ECE Report:

Import File Name	Status
PdcService.xml	SUCCESS
Customer.xml	SUCCESS
PdcEvent.xml	FAILED

Successfully Imported Files: 2

Failed to Import Files: 1

End Load ECE Report.

Handling Synchronization Failures

The most common reasons for a synchronization failure are as follows:

Copyright © 2017, 2026, Oracle and/or its affiliates.

- The billing or rating system is down
- After billing or rating system data was modified, the billing or rating system was not restarted. Hence, no data is available for synchronization with PDC
- The PDC system is down
- The transformation engines are down
- The transformation engines failed to process the PDC data and update the cross-reference database
- The billing or rating system database or the PDC database is down

Reprocessing a Failed Synchronization

To reprocess a failed synchronization:

1. Identify and fix the cause of the synchronization failure.

   For information about synchronization failures, check the **SyncPDC** log file located in the directory specified in the **<syncPDCLogFileLocation>** element of the **SyncPDCConfiguration.xml** file.

2. Run **SyncPDC**.

   See "Running SyncPDC" for more information.

   SyncPDC synchronizes the data that failed to synchronize earlier and ignores the data that was already synchronized.

Changing Display Names

When synchronizing setup components, **SyncPDC** creates display names in PDC for the corresponding data objects and their fields. You can change the default display names that **SyncPDC** creates.

To change the display name of a setup component:

1. Export the setup component to an XML file by using the **ImportExportPricing** utility.

   See "Exporting Pricing and Setup Components from PDC" for more information.

2. Open the XML file in a text editor or an XML editor.
3. Change the display name of the setup component in the XML file.
4. Save and close the file.
5. Import the updated XML file to PDC by using **ImportExportPricing**.

   See "Importing Pricing and Setup Components" for more information.

   If you later run **SyncPDC** to synchronize any changes from the billing or rating system with PDC, **SyncPDC** retains the updated component name.

Copyright © 2017, 2026, Oracle and/or its affiliates.

4

Configuring Deliverables

Learn how to use Oracle Communications Pricing Design Center (PDC) deliverables to specify how and when revenue is recognized by your company.

Topics in this document:

- About Creating Deliverables
- Configuring Deliverables
- Loading Deliverables into the PDC Database

About Creating Deliverables

A deliverable is a good or service that provides value to your customers, such as an eBook, a night in a hotel room, or a month of telephony service. In BRM, a deliverable defines the type of good or service you are selling, its standalone selling price, its revenue earning schedule, and its revenue G/L ID.

You create deliverables if your company supports deliverable-based revenue recognition. See "About Deliverable-Based Revenue Recognition" in *BRM Collecting General Ledger Data*.

The type of deliverable that you create depends on the good or service that your company is selling.

- **One-time activities**: These are discrete services that are performed by your company or by your customers, such as a company installing a service at the customer's home. All of the revenue from one-time activities is recognized at once, either at the time of purchase or when a milestone has been marked as completed.
- **Physical goods**: These are tangible products that you sell to your customers, such as DVDs, cell phones, and eBooks. All of the revenue from physical goods is recognized at once, either at the time of purchase or when a milestone has been marked as completed.
- **Metered services**: These are services with limits on how much customers can use in a cycle, such as 100 free minutes or 20 eBooks a month. The revenue from metered services is recognized on a linear earning schedule at a frequency that you specify. For example, if a customer has a one-year contract for a monthly service, your company recognizes 1/12th of the revenue at the end of each month.
- **Continuous services**: These allow customers to use an unlimited amount of a service over a specified time period, such as unlimited data for a month or unlimited access to a website for a month. The revenue from continuous services is recognized on a linear earning schedule at a frequency that you specify. For example, if a customer has a one-year contract for a monthly service, your company recognizes 1/12th of the revenue at the end of each month.

To create deliverables, do the following:

1. Ensure that deliverable-based revenue recognition is enabled in BRM. See "Enabling Deliverable-Based Revenue Recognition" in *BRM Collecting General Ledger Data*.
2. Configure your deliverables in an XML configuration file. See "Configuring Deliverables".

Copyright © 2017, 2026, Oracle and/or its affiliates.

3. Load the XML configuration file into the PDC database by using the **ImportExportPricing** utility. See "Loading Deliverables into the PDC Database".

Configuring Deliverables

To configure your deliverables, you can create your own XML configuration file that follows the format specified in the XML Schema Definition (XSD) file. Alternatively, you can make a copy of the *PDC_home***/apps/Samples/Examples/SampleDeliverable.xml** file and then customize it for your business needs.

The deliverable XML configuration file contains the following:

- A **deliverables** section, which is the top-level section that applies to all deliverable types. The elements in this section specify the deliverable's name, type, and associated G/L ID. See "About the deliverables Section".
- An **entitlements** subsection, which applies to only service-based deliverables: one-time activities, continuous services, and metered services. The elements in this subsection specify the type of service you are offering and its validity period. See "About the entitlements Subsection".
- A **revenueBasisItem** subsection, which applies to all deliverables. The elements in this subsection define the good's or service's standalone selling price and its earning schedule. See "About the revenueBasisItem Subsection".

  For physical goods, this subsection is nested under the **deliverables** section. For service-based deliverables, this subsection is nested under **entitlements**.

For information about how to configure the XML file based on the type of good or service you are selling, see the following:

- Configuring Deliverables for Physical Goods
- Configuring Deliverables for One-Time Activities
- Configuring Deliverables for Metered Services
- Configuring Deliverables for Continuous Services

About the deliverables Section

Table 4-1 describes all of the elements in the **deliverables** section.

Table 4-1 Elements in the deliverables Section

Element	Description
name	Specifies the name of the deliverable.
description	Provides a description of the deliverable.
priceListName	Set this to **Default**.
obsolete	Specifies whether the deliverable configuration is obsolete (**true**) or still active (**false**).
deliverableType	Specifies whether the deliverable is for a product or service: **PRODUCT**: Use this value for physical goods. **SERVICE**: Use this value for one-time activities, metered services, and continuous services.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 4-1 (Cont.) Elements in the deliverables Section

Element	Description
productType	Specifies the type of product: **PHYSICAL_GOODS**: Use this value for physical goods. **ADMINISTRATIVE**: Use this value for one-time activities, metered services, and continuous services. **Note**: This field is required if **deliverableType** is **PRODUCT**.
permittedType	Specifies the type of service that the deliverable can be associated with: **PRODUCT**: Use this if the deliverable can be associated with a specific service type. **CUSTOMER**: Use this if the deliverable can be associated with Account.
permittedName	If **permittedType** is **PRODUCT**, set this to the name of the service. If **permittedType** is **CUSTOMER**, set this to **Account**.
revenueGlid	Specifies the revenue G/L ID to use for tracking early termination fees.

About the entitlements Subsection

Table 4-2 describes all of the elements in the **entitlements** subsection. This subsection applies to only one-time activities, metered services, and continuous services.

Table 4-2 Elements in the entitlements Subsection

Element	Description
entitlementType	Specifies the type of service entitlement: **ALLOWANCE_ONE_TIME**: Use this for one-time activities. **FEATURE**: Use this for continuous services. **ALLOWANCE_RECURRING**: Use this for metered services.
name	Specifies the name of the service entitlement.
description	Provides a description of the service entitlement.
resourceBalElemNumCode	(ALLOWANCE_ONE_TIME and ALLOWANCE_RECURRING entitlement types only) Specifies the noncurrency balance element numeric code for tracking the service you are offering. For example, you might enter the numerical code for gigabytes for an allowance of data.
featureValue	(FEATURE entitlement types only) Describes the type of continuous service you are providing, such as unlimited data for a month or unlimited text messages for a week.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 4-2 (Cont.) Elements in the entitlements Subsection

Element	Description
allowanceValidityMode	(ALLOWANCE_ONE_TIME and ALLOWANCE_RECURRING entitlement types only) Specifies the validity period for the allowance granted to your customer: **CURRENT_BILLING**: The allowance expires after the current billing cycle. **MULTIPLE_BILLING**: The allowance expires after the specified number of billing cycles, such as 3 billing cycles. **CURRENT_RECURRING**: The allowance expires after the current frequency period, such as the current quarter or the current bimonthly period. **MULTIPLE_RECURRING**: The allowance expires after the specified number of frequency periods, such as two quarters or two bimonthly periods. **ALWAYS**: The allowance never expires. For example, if the data granted to your customer expires at the end of the billing cycle, you would enter CURRENT_BILLING.
allowanceValidityPeriod	(ALLOWANCE_ONE_TIME and ALLOWANCE_RECURRING entitlement types only) Specifies the number of cycles or periods for the allowance's validity mode. **1**: Use this if **allowanceValidityMode** is set to **CURRENT_BILLING**, **CURRENT_RECURRING**, or **ALWAYS**. *IntegerValue*: Specify the number of billing cycles or frequency periods that the allowance is valid, such as **3** for 3 billing cycles. Use this if **allowanceValidityMode** is set to **MULTIPLE_BILLING** or **MULTIPLE_RECURRING**.
allowanceValue	(ALLOWANCE_ONE_TIME and ALLOWANCE_RECURRING entitlement types only) Specifies the amount of a resource that is granted to the customer, such as 20 eBooks, 200 domestic texts, or 1 night in a hotel room.
subscriptionCycle	(ALLOWANCE_RECURRING entitlement types only) Specifies the frequency at which the allowance is granted to the customer. The supported values are: **MONTHLY**, **BIMONTHLY**, **QUARTERLY**, **SEMI_ANNUAL**, and **ANNUAL**.

About the revenueBasisItem Subsection

Table 4-3 describes all of the elements in the **revenueBasisItem** subsection.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 4-3 Elements Supported in the revenueBasisItem Subsection

Element	Description
type	Specifies the type of revenue: **RATE_BASED**: Use this for metered services and one-time activities. **VALUE_BASED**: Use this for continuous services and physical goods.
amount	Specifies the amount for the standalone selling price, such as 100, 150, or 200.
currencyBalElemNumCode	Specifies the ISO currency number for the standalone selling price, such as **840** for US Dollars or **978** for Euros.
timeUnit	Set this to **MONTH**.
timeValue	Specifies the number of months that the amount is valid, such as **2** for 2 months. The default is **1**.
milestoneType	Specifies when the revenue is recognized for physical goods: **IMMEDIATE**: Use this to recognize the revenue at the time of purchase. **FUTURE**: Use this to recognize revenue when a milestone is marked as completed. **Note**: The **milestoneType** element applies only to physical goods. However, you must enter a value for all other deliverable types to prevent an error when you load the XML configuration file.
scalable	Specifies whether the rate is scalable (**true**) or not (**false**).
quantity	(ALLOWANCE_RECURRING and ALLOWANCE_ONE_TIME entitlement types only) Specifies the quantity of the service.

Configuring Deliverables for Physical Goods

To configure a deliverable for a physical good, add the following to your XML configuration file:

- In the top-level **deliverables** section, add a name, description, and other information about your product. Ensure that you set these elements as follows:
  - Set **deliverableType** to **PRODUCT**
  - Set **productType** to **PHYSICAL_GOODS**
- In the **revenueBasisItem** subsection, do the following:
  - Set **type** to **VALUE_BASED**
  - Specify the standalone selling price for the physical good that you are selling. For example, for a standalone selling price of 20 US Dollars, you would enter:

    ```
    <amount>20.00</amount>
    <currencyBalElemNumCode>840</currencyBalElemNumCode>
    ```

  - Specify whether the revenue is recognized at purchase (IMMEDIATE) or when a milestone is marked as completed (FUTURE). For example:

    ```
    <milestoneType>IMMEDIATE</milestoneType>
    ```


Copyright © 2017, 2026, Oracle and/or its affiliates.

  - Specify the validity period for the physical good. For example, for 1 month, you would enter:

    ```
    <timeUnit>MONTH</timeUnit>
    <timeValue>1</timeValue>
    ```

The following example shows how to configure a deliverable for an Internet modem and handset with a standalone selling price of 99 US Dollars, an immediate earning schedule, and a validity period of 1 month.

```
<deliverables xmlns:cim="http://xmlns.oracle.com/communications/platform/
model/Config">
   <name>Internet Modem and Handset</name>
   <description>Internet Modem and Handset</description>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <deliverableType>PRODUCT</deliverableType>
   <productType>PHYSICAL_GOODS</productType>
   <revenueGlid>1000005</revenueGlid>
   <revenueBasisItem>
      <type>VALUE_BASED</type>
      <amount>99</amount>
      <currencyBalElemNumCode>840</currencyBalElemNumCode>
      <timeUnit>MONTH</timeUnit>
      <timeValue>1</timeValue>
      <milestoneType>IMMEDIATE</milestoneType>
      <scalable>true</scalable>
   </revenueBasisItem>
</deliverables>
```

Configuring Deliverables for One-Time Activities

To configure a deliverable for a one-time activity, add the following to your XML configuration file:

- In the top-level **deliverables** section, add a name, description, and other information about the one-time activity. Ensure that you set these elements as follows:
  - Set **deliverableType** to **SERVICE**
  - Set **productType** to **ADMINISTRATIVE**
- In the **entitlements** subsection, add a name and description, and then ensure that you set the following elements:
  - Set **entitlementType** to **ALLOWANCE_ONE_TIME**
  - Specify the noncurrency balance element numeric code for tracking the service you are offering. For example:

    ```
    <resourceBalElemNumCode>1000101</resourceBalElemNumCode>
    ```

  - Specify the validity period for the allowance. For example, for a validity period of 2 billing cycles, you would enter:

    ```
    <allowanceValidityMode>MULTIPLE_BILLING</allowanceValidityMode>
    <allowanceValidityPeriod>2</allowanceValidityPeriod>
    ```

- In the **revenueBasisItem** subsection, add the following:
  - Set **type** to **RATE_BASED**

Copyright © 2017, 2026, Oracle and/or its affiliates.

  - Specify the standalone selling price for the one-time activity. For example, for a standalone selling price of 100 Euros, you would enter:

    ```
    <amount>100.00</amount>
    <currencyBalElemNumCode>978</currencyBalElemNumCode>
    ```

The following example shows how to configure a deliverable for an Internet Modem Handset Installation with a standalone selling price of 200 US Dollars:

```
<deliverables xmlns:cim="http://xmlns.oracle.com/communications/platform/
model/Config">
   <name>Internet Modem Handset_Installation</name>
   <description>Internet Modem Handset_Installation</description>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <deliverableType>SERVICE</deliverableType>
   <productType>ADMINISTRATIVE</productType>
   <revenueGlid>1000002</revenueGlid>
   <entitlement>
      <entitlementType>ALLOWANCE_ONE_TIME</entitlementType>
      <name>Installation Entitlement</name>
      <resourceBalElemNumCode>1000101</resourceBalElemNumCode>
      <allowanceValidityMode>CURRENT_BILLING</allowanceValidityMode>
      <revenueBasisItem>
         <type>RATE_BASED</type>
         <amount>200.0</amount>
         <currencyBalElemNumCode>840</currencyBalElemNumCode>
         <timeUnit>MONTH</timeUnit>
         <timeValue>1</timeValue>
         <milestoneType>IMMEDIATE</milestoneType>
         <scalable>true</scalable>
         <quantity>1</quantity>
      </revenueBasisItem>
   </entitlement>
</deliverables>
```

Configuring Deliverables for Metered Services

To configure a deliverable for a metered service, add the following to your XML configuration file:

- In the top-level **deliverables** section, add a name, description, and other information about the metered service. Ensure that you set these elements as follows:
  - Set **deliverableType** to **SERVICE**
  - Specify the type of service that the deliverable can be associated with. For example, for telephony services, you would enter:

    ```
    <permittedType>PRODUCT</permittedType>
    <permittedName>TelcoGsmTelephony</permittedName>
    ```

- In the **entitlements** subsection, add a name and description, and then ensure that you set the following elements:
  - Set **entitlementType** to **ALLOWANCE_RECURRING**

Copyright © 2017, 2026, Oracle and/or its affiliates.

- Specify the allowance amount that is granted to the customer, such as 100 minutes or 10 gigabytes. For example, for 100 text messages, you would enter:

  ```
  <allowanceValue>100.00</allowanceValue>
  <resourceBalElemNumCode>9305</resourceBalElemNumCode>
  ```

- Specify the validity period for the allowance. For example, if a granted data allowance of 5 GB expires after 1 billing cycle, you would enter:

  ```
  <allowanceValidityMode>CURRENT_BILLING</allowanceValidityMode>
  <allowanceValidityPeriod>1</allowanceValidityPeriod>
  ```

- Specify the frequency at which the allowance is granted to the customer. For example, if a data allowance of 5 GB is granted every month, you would enter:

  ```
  <subscriptionCycle>MONTHLY</subscriptionCycle>
  ```

- In the **revenueBasisItem** subsection, add the following:
  - Set **type** to **RATE_BASED**
  - Specify the standalone selling price for the service. For example, for a standalone selling price of 100 Euros, you would enter:

    ```
    <amount>100.00</amount>
    <currencyBalElemNumCode>978</currencyBalElemNumCode>
    ```

  - Specify the validity period for the metered service. For example, for 1 month, you would enter:

    ```
    <timeUnit>MONTH</timeUnit>
    <timeValue>1</timeValue>
    ```

The following example shows how to configure a deliverable for VOIP service that grants 200 minutes per month and that has a standalone selling price of 100 US dollars:

```
<deliverables xmlns:cim="http://xmlns.oracle.com/communications/platform/
model/Config">
   <name>VoiceCallsOverIP</name>
   <description>VoiceCallsOverIP</description>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <deliverableType>SERVICE</deliverableType>
   <permittedType>PRODUCT</permittedType>
   <permittedName>Ip</permittedName>
   <revenueGlid>1000001</revenueGlid>
   <entitlement>
      <entitlementType>ALLOWANCE_RECURRING</entitlementType>
      <name>VoiceCallOverIPEntitlement</name>
      <resourceBalElemNumCode>1000101</resourceBalElemNumCode>
      <allowanceValidityMode>CURRENT_BILLING</allowanceValidityMode>
      <allowanceValidityPeriod>1</allowanceValidityPeriod>
      <allowanceValue>200</allowanceValue>
      <subscriptionCycle>MONTHLY</subscriptionCycle>
      <revenueBasisItem>
         <type>RATE_BASED</type>
         <amount>100.0</amount>
         <currencyBalElemNumCode>840</currencyBalElemNumCode>
         <timeUnit>MONTH</timeUnit>
         <timeValue>1</timeValue>
         <milestoneType>IMMEDIATE</milestoneType>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
        <scalable>false</scalable>
        <quantity>1</quantity>
      </revenueBasisItem>
   </entitlement>
</deliverables>
</cim:ConfigObjects>
```

Configuring Deliverables for Continuous Services

To configure a deliverable for a continuous service, add the following to your XML configuration file:

- In the top-level **deliverables** section, add a name, description, and other information about the continuous service. Ensure that you set these elements as follows:
  - Set **deliverableType** to **SERVICE**
  - Specify whether the deliverable can be associated with a service or an account. For example, for VOIP services, you would enter:

    ```
    <permittedType>PRODUCT</permittedType>
    <permittedName>Ip</permittedName>
    ```

- In the **entitlements** subsection, add a name and description, and then ensure that you set the following elements:
  - Set **entitlementType** to **FEATURE**
  - Specify the name and value of the continuous service you are offering. For example, for an unlimited number of minutes for VOIP calls, you would enter:

    ```
    <featureValue>UnlimitedCallMinutes</featureValue>
    ```

- In the **revenueBasisItem** subsection, add the following:
  - Set **type** to **VALUE_BASED**
  - Specify the standalone selling price for the service. For example, for a standalone selling price of 20 US dollars, you would enter:

    ```
    <amount>20.00</amount>
    <currencyBalElemNumCode>840</currencyBalElemNumCode>
    ```

  - Specify the validity period for the continuous service. For example, for 1 month, you would enter:

    ```
    <timeUnit>MONTH</timeUnit>
    <timeValue>1</timeValue>
    ```

The following example shows how to configure a deliverable for unlimited data for a month that has a standalone selling price of 100 US dollars:

```
<deliverables xmlns:cim="http://xmlns.oracle.com/communications/platform/
model/Config">
   <name>DataDownload</name>
   <description>DataDownload</description>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <deliverableType>SERVICE</deliverableType>
   <permittedType>PRODUCT</permittedType>
   <permittedName>Ip</permittedName>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
   <revenueGlid>1000001</revenueGlid>
   <entitlement>
      <entitlementType>FEATURE</entitlementType>
      <name>DataDownloadEntitlement</name>
      <featureValue>UnlimitedDataDownload</featureValue>
      <revenueBasisItem>
         <type>VALUE_BASED</type>
         <amount>100</amount>
         <currencyBalElemNumCode>840</currencyBalElemNumCode>
         <timeUnit>MONTH</timeUnit>
         <timeValue>1</timeValue>
         <milestoneType>IMMEDIATE</milestoneType>
         <scalable>true</scalable>
      </revenueBasisItem>
   </entitlement>
</deliverables>
```

Loading Deliverables into the PDC Database

To load your deliverables into the PDC database:

1. Go to the directory in which you saved your deliverable XML configuration file.
2. Run the following command:

   `ImportExportPricing -import -config` *`FileOrDirectoryName`*

   where *FileOrDirectoryName* specifies either the full path and file name of the import XML file or the directory in which the import XML files reside.

See "Importing and Exporting Pricing and Setup Components" for more information.

Copyright © 2017, 2026, Oracle and/or its affiliates.

5

Configuring Policy Specifications

Learn how to configure policy specifications for Oracle Communications Billing and Revenue Management (BRM) in Pricing Design Center (PDC).

Topics in this document:

- About Configuring Policy Specifications

About Configuring Policy Specifications

You configure policy specifications in PDC for policy-driven charging.

> **Note**
>
> Policy specifications can be used only with Oracle Communications Elastic Charging Engine (ECE).

After you configure policy specifications, you can import the policy specifications and policy labels from the customized XML files into the PDC database by running the following command:

```
ImportExportPricing -import -config FileOrDirectoryName
```

where *FileOrDirectoryName* specifies either the full path and file name of the import XML file or the directory in which the import XML files reside.

The policy specification is imported into the PDC database and transformed to BRM and ECE.

Configuring Policy Specifications

To configure policy specifications:

1. Make a copy of the *PDC_home***/apps/Samples/Examples/SamplePolicyLabel.xml** file to customize.
2. Open the file in a text editor or XML editor.
3. Add or modify the elements as required.
4. Save and close the file.
5. Make a copy of the *PDC_home***/apps/Samples/Examples/SamplePolicySpecification.xml** file to customize.
6. Open the file in a text editor or XML editor.
7. Add or modify the elements as required.
8. Save and close the file.
9. Go to the directory where you have saved the files.

Copyright © 2017, 2026, Oracle and/or its affiliates.

10. Import the policy specifications and policy labels from the customized XML files into the PDC database by running the following command:

    ImportExportPricing -import -config *FileOrDirectoryName*

The policy specifications and policy labels are imported into the PDC database and transformed to ECE.

After you configure policy specifications, you can export the policy specifications and policy labels in PDC to XML files by running the following command:

```
ImportExportPricing -export -config POLICY_SPECIFICATION
ImportExportPricing -export -config POLICY_LABEL
```

You can load the updated sample XML files into PDC by using the **ImportExportPricing** utility. See "ImportExportPricing" for the utility's syntax and parameter descriptions.

Copyright © 2017, 2026, Oracle and/or its affiliates.

6

Configuring Item Type Selectors

Learn how to configure item type selectors in Pricing Design Center (PDC) to track balances in Oracle Communications Billing and Revenue Management (BRM) product offerings.

Topics in this document:

- About Configuring Item Type Selectors
- Loading the OOB_ItemTypeSelector.XML File
- Configuring Item Type Selectors

About Configuring Item Type Selectors

You configure item type selectors in PDC so that balance impacts can be tracked appropriately for different bill items.

By default, BRM tracks balances for the following bill items: cycle arrears items, cycle forward items, cycle forward arrears items, cycle tax items, cycle incentive items, and usage items. You can load the **OOB_ItemTypeSelector.xml** file into PDC to configure the item type selectors for these bill items in PDC. See "Loading the OOB_ItemTypeSelector.XML File" for more information.

If you want to use custom bill items, you can configure item type selectors to create rules to assign balance impact to the custom bill items. You can use the **Sample_ItemTypeSelector.xml** file as a template to configure the item type selectors for custom bill items. See "About Configuring Item Type Selectors" for more information.

Table 6-1 lists the elements that are supported in the item type selector XML file, the usage of each element, and a description of how to specify each element in the XML file.

Table 6-1 Elements Supported in the Item Type Selector XML File

Element	Syntax	Description
itemTypeSelectors	`<itemTypeSelectors>`	The root element of **ItemTypeSelector.xml**.
name	`<name>`*`ItemTypeSelectorName`*`</name>`	*ItemTypeSelectorName* specifies the item type selector name.
description	`<description>`*`ItemTypeSelectorDescription`*`</description>`	*ItemTypeSelectorDescription* contains the description of the item type selector.
priceListName	`<priceListName>`*`PriceListName`*` </priceListName>`	*PriceListName* specifies the price list name. BRM uses only one internal price list. The only value is **Default**.
applicableToName	`<applicableToName>`*`ApplicableToName`*`</applicableToName>`	*ApplicableToName* specifies the service name or **Account**. The attributes of this service or account are used in the rules.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 6-1 (Cont.) Elements Supported in the Item Type Selector XML File

Element	Syntax	Description
eventSpecName	`<eventSpecName>EventName</eventSpecName>`	*EventName* specifies the name of any event. The attributes of this event are used in the rules.
itemGroup	`<itemGroup>ItemGroupName</itemGroup>`	Specifies the item group to be used for aggregating the balance impacts, where *ItemGroupName* specifies the name of the item group. The **<itemGroup>** element is associated with the **<ItemTag>** element in BRM. To aggregate balance impacts based on the item configuration value in the business profile, specify the item configuration value as *ItemGroupName*. See "Improving Performance by Using Multiple Item Configurations" in *BRM System Administrator's Guide* for more information on item configuration. **Note:** If you specify the item configuration value as *ItemGroupName*, Oracle recommends that you do not change that value in the business profile. The default value is **Default**.
applicableToAllBalImpact	`<applicableToAllBalImpact>BalanceImpactsOption</applicableToAllBalImpact>`	Specifies whether this item type selector is applicable to all balance impacts, where *BalanceImpactsOption* is: • **true** if the item type selector applies for all the balance impacts • **false** if the item type selector applies only to specific balance impacts
rule	`<rule>` `<ruleName>RuleName</ruleName>` `<ruleOrder>RuleOrder</ruleOrder>` `<expression>Expression</expression>` `<specName>ItemSpecificationName</specName>` `</rule>`	Contains the rules for assigning the balance impacts to the bill items, where: • *RuleName* specifies the rule name. • *RuleOrder* specifies the order in which the rules must be applied. • *Expression* contains the details to derive which items to assign balance impacts to. **Note:** You can add multiple rules in the item type selector. Each rule points to the item specification in the same item type selector. Each rule in the item type selector can refer to a different attribute. For example, Rule 1 can use an event attribute, Rule 2 can use an attribute of the charging result, and Rule 3 can use a service attribute.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 6-1 (Cont.) Elements Supported in the Item Type Selector XML File

Element	Syntax	Description
expression	<expression> <separator>*Delimiter*</separator> <operation>*Operator*</operation> <fieldName> *FieldName*</fieldName> <fieldKind>*FieldKind*</fieldKind> <fieldValue>*FieldValue*</fieldValue> </expression>	Contains the details to derive which items to assign balance impacts to. • *Delimiter* specifies the character that is used to separate the field values. • *Operator* specifies the operator to be used in the expression. The valid values are: **EQUAL_TO** **NOT_EQUAL_TO** **GREATER_THAN** **GREATER_THAN_EQUAL** **LESS_THAN** **LESS_THAN_EQUAL** **IN_LIST** **REGEX** • *FieldName* specifies the field name. If you specify **CHARGING_RESULT_SPEC** as the *FieldKind*, the valid values are: **TAX_CODE** **GLID** **RUM** **BALANCE_ELEMENT** **ZONE_RESULT** **TIMEMODEL_TAG_NAME** **GENERIC_SELECTOR_RESULT** **ORIG_ZONE_RESULT** **PRICING_NAME** **TIME_MODEL_NAME** **ZONE_MODEL_NAME** **EVALUATED_ZONE_MODEL_NAME** **EVALUATED_ZONE** **CHARGE_RATE_PLAN_NAME** • *FieldKind* specifies the type of attributes to be selected for assigning the balance impacts. The valid values are: **PRODUCT_SPEC_FIELD** **EVENT_SPEC_FIELD** **CUSTOMER_SPEC_FIELD** **CHARGING_RESULT_SPEC** • *FieldValue* specifies the field value that is used for tracking balance impacts.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 6-1 (Cont.) Elements Supported in the Item Type Selector XML File

Element	Syntax	Description
itemSpec	<itemSpec> <name>*ItemSpecficationName*</name> <description>*ItemSpecificationDescription*</description> <priceListName>*PriceListName* </priceListName> <type>*ItemType*</type> <default>*DefaultOption*</default> <aggregationType>*AggregationType*</aggregationType> <category>*ChargeCategory*</category> </itemSpec>	Contains the details about the item specification, where: • *ItemSpecficationName* specifies the item specification name. The item specification name is associated with the item tag field in BRM. • *ItemSpecificationDescription* contains the description of the item specification. • *PriceListName* specifies the price list name. The valid value is **Default**. • *ItemType* specifies the type of the bill item for assigning balance impacts. This must match the item type created in BRM. • *DefaultOption* specifies whether an item specification is the default specification for the bill item. The valid values are **true** and **false**. • *AggregationType* specifies whether the item accumulates charges or tracks each charge separately. The valid values are: **CUMULATIVE_NONPRECREATE**, which specifies that the bill item accumulates charges throughout the billing cycle and the bill item need not be pre-created in BRM **INDIVIDUAL_NONPRECREATE**, which specifies that the bill item stores a charge for a single event and the bill item need not be pre-created in BRM **CUMULATIVE_PRECREATE**, which specifies that the bill item accumulates charges throughout the billing cycle and the bill item needs to be pre-created in BRM • *Category* specifies the charge category. See "Configuring Charges in Charge Offers" for more information. **Note:** You can add multiple item specifications, but you can add only one default item specification. You can use the same item specification in multiple item type selectors, but the item specification name must be unique.

Loading the OOB_ItemTypeSelector.XML File

To load the **OOB_ItemTypeSelector.xml** file:

1. If you have created custom bill items in BRM, make a backup copy of the customized **config_item_tags.xml** and **config_item_types.xml** files in BRM.
2. Go to the *PDC_home***/apps/Samples/Examples** directory.
3. Import the default billing item assignments from the **OOB_ItemTypeSelector.xml** file into the PDC database by running the following command:

```
ImportExportPricing -import -config /OOB_ItemTypeSelector.xml -ow
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

The item type selector is imported into the PDC database and the existing item tag-to-item type mapping in the BRM database is overwritten.

Configuring Item Type Selectors

To configure an item type selector:

1. Create the custom bill item in the database by subclassing the **/item** storable object. For example, you can create an **/item/national** object for tracking the charges for national calls. See "Creating Custom Fields and Storable Classes" in *BRM Developer's Guide*.
2. Make a copy of the *PDC_home***/apps/Samples/Examples/Sample_ItemTypeSelector.xml** file to customize.
3. Open the file in a text editor or XML editor.
4. Add or modify the elements as required. See Table 6-1 for the usage and description of each element.
5. Save and close the file.

> **Note**
>
> You can save the file with a different name and location or use the original file.

6. Go to the directory where you have saved the file.
7. Import the item type selector from the customized XML file into the PDC database by running the following command:

   ```
   ImportExportPricing -import -config FileOrDirectoryName
   ```

   where *FileOrDirectoryName* specifies either the full path and file name of the import XML file or the directory in which the import XML files reside.

   The item type selector is imported into the PDC database and transformed to BRM and Oracle Communications Elastic Charging Engine (ECE).

Copyright © 2017, 2026, Oracle and/or its affiliates.

7

Configuring Value Maps

Learn how to configure value maps in Pricing Design Center (PDC) to group event attributes into manageable categories in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

- About Value Maps
- Loading the Sample Value Map into the Database
- Configuring Value Maps Using SampleValueMap.XML

About Value Maps

You create value maps to associate zone names with values that apply to a single event attribute, such as an area code or an IP address. This allows you to set up services for location-based charging, which charges your customers based on their location.

You can configure value maps in PDC using the **SampleValueMap.xml** file and **ImportExportPricing** utility. For custom zones, use the **SampleValueMap.xml** as a template to map your zone names with a list of event attributes.

Table 7-1 lists the elements that are supported in the sample value map XML file, the usage of each element, and a description of how to specify each element in the XML file.

Table 7-1 Elements Supported in the Sample Value Map XML File

Element	Syntax	Description
valueMap	<valueMap>	The root element of the **SampleValueMap.xml** file.
name	<name>*valueMapName*</name>	*valueMapName* specifies the name of the value map.
description	<description>*valueMapDescription*</description>	*valueMapDescription* specifies the description of the value map.
priceListName	<priceListName>*PriceListName*</priceListName>	*PriceListName* specifies the price list name. BRM uses only one internal price list. The only value is **Default**.
obsolete	<obsolete>*BooleanValue*</obsolete>	Boolean flag to obsolete the promoted objects.
exactMatch	<exactMatch>*BooleanValue*</exactMatch>	Boolean value. When **true**, the event attribute when compared with *keyName* or *valueMapValue*, must be an exact match. If **false**, the event attribute contains *keyName* or *valueMapValue*.
validityPeriod	<validityPeriod>	Root key of the validity period.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 7-1 (Cont.) Elements Supported in the Sample Value Map XML File

Element	Syntax	Description
validFrom	<validFrom>*value*</validFrom>	*Value* describes the start time for the value map. If *value* is set to zero, there is no start time. Otherwise, it is in the format of *YYYYMMDD* (For example: 20240131).
valueMapKey	<valueMapKey>	**valueMapKey** contains the *keyName* and 0 or more *valueMapKey entries.*
keyName	<keyName>*keyName*</keyName>	*keyName* describes the group name for the list of *valueMapValue* entries. This is a required field.
valueMapValue	<valueMapValue>*value*</valueMapValue>	*value* describes the value to be compared. This is a required field.

Loading the Sample Value Map into the Database

To load the **SampleValueMap.xml** file:

1. Go to the *PDC_home***/apps/Samples/Examples** directory.
2. Import the **SampleValueMap.xml** file into the PDC database by running the following command:

```
ImportExportPricing -import -config /SampleValueMap.xml -ow
```

The sample value map is imported into the PDC database and the existing mapping in the BRM database is overwritten.

Configuring Value Maps Using SampleValueMap.XML

To configure value maps:

1. Make a copy of the *PDC_home***/apps/Samples/Examples/SampleValueMap.xml** file to customize.
2. Open the file in a text editor or XML editor.
3. Add or modify the elements as required. See Table 7-1 for the usage and description of each element.
4. Save and close the file.

> **Note**
>
> You can save the file with a different name and location or use the original file.

5. Go to the directory where you have saved the file.

Copyright © 2017, 2026, Oracle and/or its affiliates.

6. Import the value map from the customized XML file into the PDC database by running the following command:

   ```
   ImportExportPricing -import -config FileOrDirectoryName
   ```

   where *FileOrDirectoryName* specifies either the full path and file name of the import XML file or the directory in which the import XML files reside.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Part II

Creating Pricing Components

This part describes how to create Oracle Communications Billing and Revenue Management (BRM) pricing components in Pricing Design Center (PDC).

This part contains the following chapters:

- Configuring Charge Offers
- Configuring Charges in Charge Offers
- Configuring Pricing in Charge Offers
- Pricing Configurations in Charge Offers
- Creating Discount Offers
- Creating Subscription Terms
- Creating Bundles
- Creating Packages and Package Lists
- Sharing Charges and Discounts
- Creating Configurable Recurring Offers

Copyright © 2017, 2026, Oracle and/or its affiliates.

8

Configuring Charge Offers

Learn how charge offers function in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

- About Charge Offers and Charges
- Configuring Charge Offer Usage and Ownership
- Applying Recurring Charges on a Specific DOM
- Configuring Charge Offers and Revenue Recognition
- Restricting the End Time of Granted Balances That Start on First Usage
- Configuring How Pricing Configurations for Date Ranges are Applied
- Charge Offers and General Ledger
- Configuring Full Day Proration
- Configuring Deferred Billing in Charges

About Charge Offers and Charges

Charge offers determine the price of one or more events associated with a service. For example:

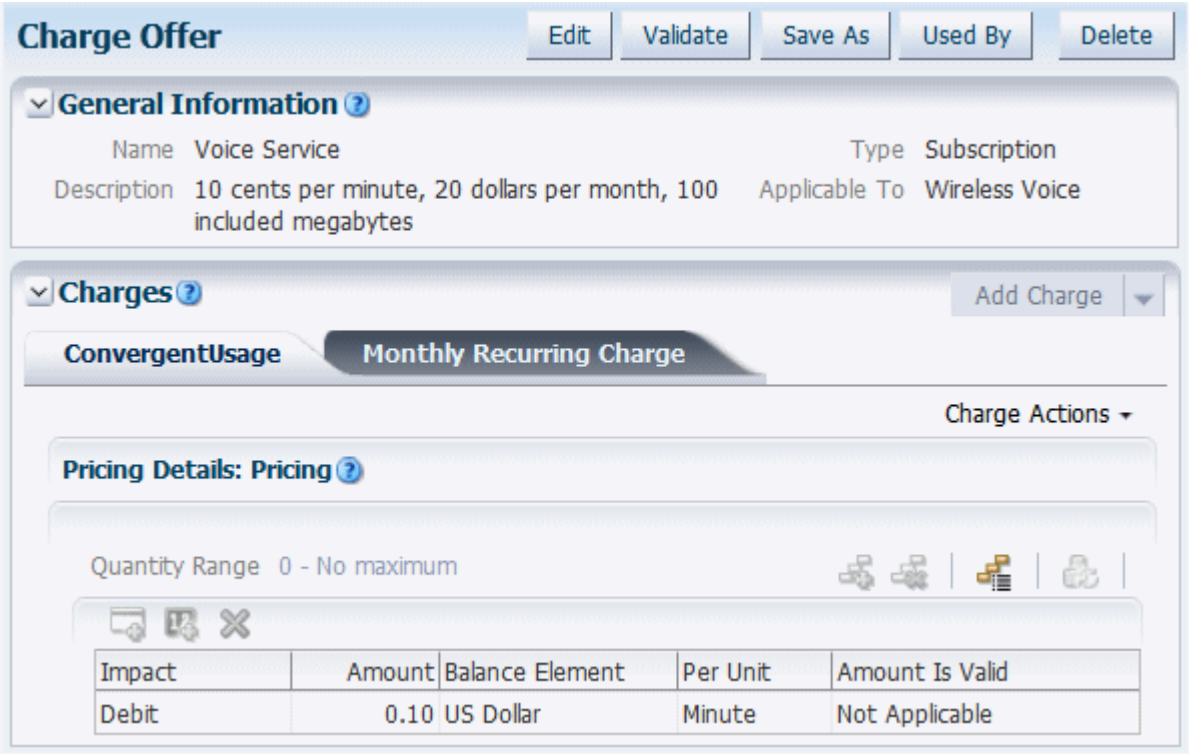
- For a mobile phone service:
  - $0.10 cents per call on peak time
  - $0.5 cents per call on off-peak time
  - $20 setup fee
- For a video download service:
  - $10 monthly fee
  - $1 fee per video download

A charge offer includes one or more charges. Figure 8-1 shows a charge offer in PDC. This charge offer includes two charges:

- ConvergentUsage charges 10 cents per minute for voice usage.
- Monthly Recurring Charge charges a 20-dollar monthly fee, and credits 100 minutes per month.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Figure 8-1 Charge Offer in PDC



Each charge can have one or more balance impacts. In Figure 8-1 the balance impact for usage is 10 cents per minute. The charge is determined primarily by the following columns in the balance impact in PDC:

- **Impact:** Can be a debit or a credit.
- **Amount:** The amount of the balance impact.
- **Balance Element:** The type of balance that is impacted. For a debit impact, this is usually a currency balance. For a credit impact, this is usually a noncurrency balance.
- **Per Unit:** What the charge is based on; for example, number of megabytes downloaded or number of minutes used.

The **Per Unit** value is typically the RUM that the charge is based on, or a measurement related to it. For example, if the RUM used to rate the event is seconds, the **Per Unit** value can be seconds, minutes, or hours.

For usage charges, you can also use the following options for the **Per Unit** value:

- **Fixed Charge:** Use this option to charge a flat fee for the event. Fixed charges are unaffected by the quantity of the event.
- **Amount Used:** This bases the charge on the quantity of the event is measured. For example, to charge for SMS messages, each event is one message, so **Amount Used** is always one message. If the **Amount** column is **.10** and the **Per Unit** column is **Amount Used**, the charge is 10 cents for each SMS message.

Figure 8-2 shows a balance impact that charges for the amount used. This balance impact could be used for charging an SMS service, a video download, and so on.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Figure 8-2 Balance Impact With No RUM Specified

Impact	Amount	Balance Element	Per Unit
Debit	0.50	US Dollar	Amount Used

The options available in a balance impact depend on the charge offer type and the charge type. For example, the **Per Unit** value is not used for one-time charges such as purchase fees, or for recurring charges. In those cases, the event is always charged the same amount.

Figure 8-3 shows the balance impacts for a charge that charges a recurring $20 monthly fee, and credits 500 minutes every month.

Figure 8-3 Balance Impacts for Monthly Fee and Monthly Megabyte Credit

Impact	Amount	Balance Element
Debit	20.00	US Dollar
Credit	500.00	Included Megabytes

When you create a charge offer, you do the following:

1. Create the charge offer in PDC. In the charge offer, you define charge offer attributes such as the service that the charges apply to, how a customer can purchase and own the charge offer, tax settings, and so on. See "Configuring Charge Offer Usage and Ownership".

   If you configured product specification attributes for charge offers in XML templates, you can set their values. See "Configuring Product Specification Attributes for Pricing Components".
2. Create one or more charges. For each charge, you define the type of charge; for example, a monthly fee or a usage fee, how to measure the charge, and other properties. See "Configuring Charges in Charge Offers".
3. Define the pricing for each charge. The pricing specifies one or more balance impacts, and how to apply the balance impacts; for example, based on the quantity of the event. See "Configuring Pricing in Charge Offers".

Configuring Charge Offer Usage and Ownership

When you create a charge offer, you associate it with one of the following:

- **A single service:** For example, to charge for GSM usage, associate the charge offer with the GSM telco service.
  Multiple charge offers can be associated with the same service.
- **No service:** In this case, you associate the charge offer with **Account**. You might create a charge offer for late charges or for coupons.
  You cannot charge for usage events with a charge offer set to **Account**.

When you create a charge offer, you specify the charge offer type and the settings that control how the charge offer can be purchased and owned. The *charge offer type* determines how the charge offer can be used:

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **Subscription:** These charge offers are purchased and owned by subscribers, and the charges apply only to the charge offer owner. Most charge offers are subscription charge offers. They can charge for any event type.
- **Item:** These charge offers contain charges that are applied only once, for example, a purchase fee or cancel fee. Item charge offers cannot include recurring charges or usage changes.
- **System:** These charge offers apply to all subscribers who use a particular service. System charge offers are not owned by subscribers and are exclusively for usage charges. They cannot be used with subscription charges such as purchase or recurring events.

In addition to defining the service and the charge offer type, you can define the following settings:

- When the charge offer can be purchased, which can be immediately, or in a specified date range.

  > **Note**
  >
  > To be added to a bundle, a charge offer must have a purchase period that is the same as or greater than the bundle's purchase period.

- The validity period for the charge offer. For example, the validity period specifies when the customer can use the service that the charge offer applies to. The validity period is not set in the charge offer itself. It is set when you configure the bundle that the charge offer belongs to. See "About the Validity Periods of Offers in Bundles".
- The provisioning tag to associate with the charge offer.

  You must create provisioning tags in BRM before you can assign them in charge offers. See "Creating Provisioning Tags" in *BRM Provisioning Services*.
- The policy specification to associate with the charge offer.

  You must configure policy specifications in PDC before you can assign them in charge offers. See "Configuring Policy Specifications".
- The tax supplier that will collect the taxes.

  You need to define tax suppliers in BRM before you can assign them in charge offers. See "Creating Tax Suppliers" in *BRM Calculating Taxes*.
- The charge offer priority. When multiple charge offers apply to the same service-event pair, BRM can consider the charge offers in the order of their priority. If a portion of the event remains unrated after BRM applies all the charges in the charge offer with the highest priority, each subsequent priority charge offer is analyzed until the entire event is rated.
- How the charge offer can be owned or purchased.
  - **Owned:** For example, if a charge offer provides an email service, you might want to limit the number of email login names that a customer can own.
  - **Purchased:** For example, if a charge offer includes an item such as a t-shirt, you might want to limit the number of t-shirts that can be purchased simultaneously.
  - **Partially purchased:** For example, if a charge offer gives customers 300 minutes of off-peak calls for $30, customers might be permitted to purchase half that amount for half the price.
- Whether this charge offer can be automatically shared with all members of a billing hierarchy.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- Whether to send notification messages to your customers when their subscription that contains this charge offer is about to expire or renew.
- The tax exemption selector to apply to the charge offer.

When you create a charge offer, to ensure a full day proration, set the rounding values for the following:

- **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.

You configure charge offer settings by using one of the following:

- The PDC UI. For more information, see "Creating a Charge Offer" in *PDC Online Help*.
- The **ImportExportPricing** utility. For information about how to configure charge offer settings using the utility and an XML file, see the sample charge offer XML file (*PDC_home***/apps/Samples/Examples/SampleChargeOfferWithPSCAttributes.xml**), "Charge Offer Configuration Examples", and "ImportExportPricing".

Applying Recurring Charges on a Specific DOM

By default, all recurring charges in a charge offer are aligned with your customer's billing day of month (DOM). For example, if customers have a billing DOM of the 1st and purchase a charge offer on January 10, the recurring charge is prorated and applied on January 10 for the interval January 10 to February 1. For subsequent cycles, the recurring charge is applied on the billing date (for the interval February 1 to March 1, March 1 to April 1, and so on).

You can configure PDC to instead apply recurring charges on a specific DOM. For example, apply recurring charges on the 5th of each month, regardless of the billing or purchase date. This is useful for charge offers with validity that is defined in hours, minutes, and seconds rather than days, weeks, and months.

You can configure PDC to apply recurring charges on a specific DOM by using the following:

- The PDC UI. To do so, use the **Cycle Alignment Day of Month** field in the charge offer settings. See "Specifying Charge Offer Settings" in *PDC Online Help*.
- The **ImportExportPricing** utility. To do so, in your input XML file, set the **purchaseCycleDOM** element under the **chargeOffering** element. See "Aligning Recurring Charges and Product Validity to a Specific Day of the Month".

Configuring Charge Offers and Revenue Recognition

When you create a charge offer, you specify how your company will recognize the revenue that is associated with it. Revenue recognition is an accounting principle that specifies when your company can recognize the money it receives from the products and services that it sells to customers.

If your company uses deliverable-based revenue recognition and the charge offer contains a recurring or one-time charge, add one or more deliverables to the charge offer. See "About Deliverable-Based Revenue Recognition" in *BRM Collecting General Ledger Data* for more information.

Copyright © 2017, 2026, Oracle and/or its affiliates.

If your company uses event-based revenue recognition, leave this section blank. See "About Event-Based Revenue Recognition" in *BRM Collecting General Ledger Data* for more information.

Restricting the End Time of Granted Balances That Start on First Usage

You can configure BRM to automatically restrict the validity period of granted noncurrency balances to end no later than the end time of the charge offer or discount offer that grants the balance.

When you configure balance 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 offer. Restricting the balance end time to the offer's end time ensures that the balance cannot continue to be consumed after the offer expires.

> **Note**
>
> When an offer is *canceled*, the validity period end time of balances granted by that charge offer or discount offer is set to the time of the cancellation.

When you restrict balance validity end time, BRM sets the end time to the offer's end time at the time of the grant. When the customer consumes the granted balance for the first time, the relative end time is calculated:

- If the calculated end time is *later* than the granting offer's end time, the balance validity period uses the offer's end time.
- If the calculated end time is *earlier* than the granting offer's end time, the balance validity period uses the calculated end time.

To enable this feature, run the **pin_bus_params** utility to change the **RestrictResourceValidityToOffer** business parameter. See "pin_bus_params" in *BRM Developer's Guide* for more information about this utility.

To restrict balance validity end times to charge offer or discount offer end times:

1. Go to *BRM_home***/sys/data/config**.
2. Create an XML file from the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsMultiBal bus_params_multi_bal.xml
   ```

3. In the file, change **FALSE** to **TRUE**:

   ```
   <RestrictResourceValidityToOffer>FALSE</RestrictResourceValidityToOffer>
   ```

4. Save the file as **bus_params_multi_bal.xml**.
5. Load the XML file into the BRM database:

   ```
   pin_bus_params bus_params_multi_bal.xml
   ```

6. Stop and restart the CM.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Configuring How Pricing Configurations for Date Ranges are Applied

When adding a charge to a charge offer, you can specify that different pricing configurations are effective during different date ranges by adding date ranges to the balance impacts. On the charge offer, you can configure how these different pricing configurations are chosen.

You can decide whether customers who subscribe during one date range will keep the same pricing configuration after that range ends or move to the new pricing configuration. You can also determine whether the pricing configuration is chosen based on the service instantiation date or the purchase date.

This is useful when you add new pricing configurations with new date ranges after customers have already subscribed to a charge offer.

You specify what happens after one date range ends in the charge offer settings. Existing subscriptions can:

- Move to the pricing configuration for the next date range, with the balance impacts configured for the next date range. This is the default behavior.
- Continue to use the old pricing configuration, with the balance impacts configured for the old date range. With this option, you are effectively maintaining separate versions of the same charge offer for different customers.
  You can configure BRM to determine the pricing configuration for new subscriptions based on:
  - The purchase date: Any subscriptions purchased within the new date range will use the new pricing configuration.
  - The service instantiation date: Any new services instantiated during the new date range will use the new pricing configuration. This option is useful if your services often have a long delay between purchase and instantiation date. It allows customers who purchase at the old price to take advantage of the new price if it comes into effect before their new service is available.

See "Configuring Effective Dates for Pricing" for more information about configuring date ranges.

The following examples describe how these configurations determine the pricing configuration that applies to existing and new subscriptions. In the examples, you have an existing charge offer with a balance impact that credits 1 GB of free data every week. You add a new balance impact crediting 1.5 GB, set the end date of the old balance impact to June 10th, and set the start date for the new balance impact to June 11th. Depending on your charge offer settings, the following happens:

- If you configured the charge offer to move to the new pricing, all subscriptions get 1 GB until June 10th, and 1.5 GB thereafter.
- If you configured the charge offer to use the old pricing for existing subscriptions, but determine pricing for new subscriptions based on the purchase date, subscriptions purchased before June 10th continue to get 1 GB, even for events occurring after June 10th. Subscriptions purchased after June 10th get 1.5 GB.
- If you configured the charge offer to use the old pricing for existing subscriptions, but determine pricing for new subscriptions based on the service instantiation date, subscriptions instantiated before June 10th continue to get 1 GB, even for events occurring

Copyright © 2017, 2026, Oracle and/or its affiliates.

after June 10th. Subscriptions instantiated after June 10th get 1.5 GB, even if they were purchased before.

You can also make these configurations in XML when importing pricing components using the **ImportExportPricing** utility. See "Using Date Ranges for Versioning".

Charge Offers and General Ledger

You assign a general ledger ID (G/L) ID to balance impacts. This enables you to track revenue for each type of charge; for example, you can assign different G/L IDs for monthly fees and usage charges.

Configuring Full Day Proration

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

- **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.

You can configure full day proration using the following:

- The PDC UI. See "Offer Settings Subsection" in *PDC Online Help*.
- The **ImportExportPricing** utility. To do so, in your input XML file, set the <**validityRounding**> and <**scaleRounding**> elements under the <**ChargeOffering**> element. See "Setting Full Day Proration".

Configuring Deferred Billing in Charges

Deferred billing enables you to charge customers for services after the service period has ended. For each charge, you can configure how long after the service you will bill the customer. For example, if a customer purchases an offer on May 1 and the offer's $10 monthly fee is deferred by two months, the customer does not pay the fee for the first two months. On July's bill, the customer is charged $10 for May's service. In the August bill, the customer is charged $10 for June's service. From an accounting perspective, each charge applies to the service period that occurred two months prior, rather than the month in which the fee is billed.

> **Note**
>
> Deferred billing (in-arrears billing) is supported only for cycle forward events.

Copyright © 2017, 2026, Oracle and/or its affiliates.

9

Configuring Charges in Charge Offers

Learn how to configure charges in your Oracle Communications Billing and Revenue Management (BRM) charge offers.

Topics in this document:

- About Charges
- Specifying When a Charge is Effective
- Specifying Whether to Charge Inactive, Canceled, or SuspendedActive Accounts
- Configuring Taxation in Charges
- Configuring In-Advance Billing in Charges
- Configuring Cycle Alignment for Recurring Charges
- About Prorating Recurring Charges and Rollovers
- About Creating Charge Offers with Over 500 Charges

About Charges

Each charge in a charge offer has a charge category and a charge type. The charge category determines which events the charge can be configured for:

- **Usage events:** Charges for using a service, such as telephone calls or broadband sessions.
- **Recurring events:** Ongoing charges that are not generated or affected by usage, such as a monthly subscription fee.
- **One-time events:** Nonrecurring charges, such as setup or cancellation fees.
- **Rollover events:** Charges that extend the validity of unused balances to succeeding cycles. For example, included minutes are often rolled over.
- **Fold events:** Charges used to zero-out a balance or convert one balance into another. For example, you could configure a fold charge to zero-out unused included hours at the end of each month or convert frequent flyer miles to a dollar amount. Fold charges must be associated with an existing charge selector whose event matches the fold event (see "About Selectors").

The charge type depends on the charge category. For example:

- If the charge category is **Recurring**, the charge type could be monthly or yearly.
- If the charge category is **Usage**, the charge type could be voice usage or data download.

In addition to the charge category and charge type, you define the following:

- The pricing profile to use with the charge. Pricing profiles specify the pricing features that are displayed in the PDC UI. See "Working with Profiles" for more information.
- The RUM to measure the event, such as **Duration**, **Occurrence**, or **Messages**. See "Configuring Ratable Usage Metrics (RUMs)" for more information.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- Whether to continue or stop applying charges to accounts with an Inactive status, Canceled status, or Suspended Active state. See "Specifying Whether to Charge Inactive, Canceled, or SuspendedActive Accounts" for more information.
- The currency specified for a charge, such as U.S. dollars or euro. This currency defines the currency balance element you can select when configuring pricing for the charge.
- How and when to apply taxes. See "Configuring Taxation in Charges" for more information.
- The dates during which the charge is effective. By default, this period starts immediately and never ends. See "Specifying When a Charge is Effective" for more information.
- Settings for recurring charges and rollovers. See:
  - Configuring In-Advance Billing in Charges
  - Configuring Cycle Alignment for Recurring Charges
  - About Prorating Recurring Charges and Rollovers

Specifying When a Charge is Effective

You can apply a date range to a charge. For example, you can set a fixed start date and an end date or specify a date range relative to when the charge offer was purchased. If you include multiple charges in a charge offer, you can configure many date ranges for each charge.

If you want to apply the same charge with multiple variations applied in different date ranges, you can use multiple date ranges in a single charge. You can use different pricing for each date range. For example, you could create a charge of $10 for the first date range and $30 for the second.

BRM supports the following types of charge date ranges:

- **Fixed:** Specifies a period that starts and ends on particular dates. For example:
  - Immediately through 6/1/2012
  - 6/1/2012 through 1/1/2013
  - 1/1/2013 through never ends

  Fixed date ranges cannot overlap.
- **Relative:** Begins at a time relative to the time the charge is purchased and continues for a specified length of time, such as days, hours, minutes, or seconds. The purchase date is the day the charge offer is added to the account. Unlike fixed date ranges, relative date ranges can overlap.

> **Note**
>
> In addition to specifying date ranges for charges, you can specify date ranges for the balance impacts in a charge. See "Configuring Effective Dates for Pricing" for information.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Specifying Whether to Charge Inactive, Canceled, or SuspendedActive Accounts

By default, PDC applies charges to only active accounts with an active service. However, you can also configure PDC to apply charges to accounts with:

- An **Inactive** status.
- A **Cancelled** status.
- A **SuspendedActive** custom life cycle state. This option is valid only if your system includes a custom life cycle state named **SuspendedActive**.

> **Note**
>
> These settings affect the rerating process. If you specify not to charge accounts with an **Inactive**, **Cancelled**, or **SuspendedActive** status, charges are not applied to those accounts during the rerating process. If you specify to charge accounts with an **Inactive**, **Cancelled**, or **SuspendedActive** status, charges are applied only to **Active** accounts during the rerating process.

To configure BRM to apply charges to **Inactive**, **Cancelled**, or **SuspendedActive** accounts:

1. Enable the **ApplyChargeOnInactiveOrCancelledProduct** business parameter by doing the following:
   a. Go to *BRM_home*/**sys/data/config**.
   b. Create an XML file from the **/config/business_params** object:

      ```
      pin_bus_params -r BusParamsSubscription bus_params_subscription.xml
      ```

   c. In the file, set **<ApplyChargeOnInactiveOrCancelProduct>** to **enabled**:

      ```
      <ApplyChargeOnInactiveOrCancelProduct>enabled</
      ApplyChargeOnInactiveOrCancelProduct>
      ```

   d. Save the file as **bus_params_subscription.xml**.
   e. Load the XML file into the BRM database:

      ```
      pin_bus_params bus_params_subscription.xml
      ```

   f. Stop and restart the CM.
2. Create a custom life cycle state named **SuspendedActive**. See "Creating Custom Service Life Cycles" in *BRM Managing Customers*.
3. When you create a charge, specify whether to apply the charge to accounts with an **Inactive** status, **Cancelled** status, or **SuspendedActive** state. To do so, use one of the following:
   - PDC.

     When adding a charge, set the **Stop Charging** list in the Create Charge dialog box. For more information, see "Adding a New Charge" in *PDC Online Help*.
   - The **ImportExportPricing** utility.

Copyright © 2017, 2026, Oracle and/or its affiliates.

For each charge defined in your import XML file, set the **<validIfCancelled>**, **<validIfInactive>**, or **<validIfSuspendedActive>** element to **true** *not* to apply charges or to **false** to apply charges. For example, the following specifies *not* to apply charges to accounts with a **Cancelled** status and to apply charges to accounts with an **Inactive** status or a **SuspendedActive** state:

```
<chargeOffering xmlns:pdc="http://xmlns.oracle.com/communications/
platform/model/pricing" externalID="externalID">
   <name>co1</name>
   <internalId>4ae7317b-ba27-49f1-aa45-ab00710b417a</internalId>
   <pricingProfileName>Product Offering</pricingProfileName>
   ...
   <chargeEventMap>
      <eventName>EventBillingCycleDiscount</eventName>
      <validIfCancelled>true</validIfCancelled>
      <validIfInactive>false</validIfInactive>
      <validIfSuspendedActive>false</validIfSuspendedActive>
      ...
   </chargeEventMap>
</chargeOffering>
```

For more information, see "Importing and Exporting Pricing and Setup Components".

> **Note**
>
> When **ApplyChargeOnInactiveOrCancelProduct** is enabled, BRM continues applying charges to inactive or canceled products so that those products are considered during rating.

Configuring Taxation in Charges

You can specify when and how to calculate taxes for each charge in a charge offer. In each charge, you set the following:

- When to apply taxes:
  - **Event time**: Taxes are calculated when the event is rated and then added to the customer's account balance. This way, you always have an accurate reading of a customer's account balance at any time in the accounting cycle.
  - **Billing time**: Taxes are applied to the charge when the customer's bill is generated. Deferring tax calculation to the billing process reduces rounding errors because all events of the same type are calculated together. For example, taxes are calculated on the total amount of usage fees rather than on individual usage events.
  - **Dynamically**: Taxes are calculated at billing time using the tax rate at the time the event occurred. This provides the benefits of billing-time taxation while allowing you to change tax rates in the middle of a billing cycle. For example, assume a recurring charge's tax rate changes from 2% to 3% on May 15. If a billing cycle ends on May 30, BRM would use the 2% rate to calculate the charge's April 30 through May 14 taxes and the 3% rate to calculate the charge's May 15 through May 29 taxes.
  - **None**: Do not apply taxes.
- How to apply taxes:

Copyright © 2017, 2026, Oracle and/or its affiliates.

- Using a specified tax code to calculate the tax amount.
- Using a tax selector, which assigns a tax code based on account, service, event, and service attributes.

> **Note**
>
> You need to define tax codes before you can assign them to charges. For information, see "Creating Tax Codes" in *BRM Calculating Taxes*.

You can configure the taxes to apply and when to calculate taxes by using the following:

- The PDC UI. To do so, use the Taxation section of the Create Charge dialog box. See "Taxation Section" in *PDC Online Help*.
- The **ImportExportPricing** utility. To do so, in your input XML file, set the **<taxTime>** and **<taxCode>** elements under the **<ChargeRatePlan>** element. See "Setting Taxation for Charges".

Configuring In-Advance Billing in Charges

In-advance billing enables you to charge customers in advance on the first bill. For each charge, you can configure how far in advance you will bill the customer. For example, if a customer purchases an offer on May 1 and the offer's $10 monthly fee is billed three months in advance, the total charge in the customer's first bill is $30. The total charge in the next (June) bill is $10, but from an accounting perspective, the $10 fee applies to August, not to June.

> **Note**
>
> In-advance billing is supported only for cycle forward events, not cycle arrears or cycle forward arrears events.

Configuring Cycle Alignment for Recurring Charges

Recurring charges are based on BRM system events that occur on cycles. You can apply recurring charges in the following ways:

- On the charge offer purchase date. For example, suppose the billing date is the 1st of the month, and the charge offer is purchased on January 10. In that case, the charge is applied on the 10th of every month (from January 10 to February 10, February 10 to March 10, and so on).
- On the customer's current billing date (default). Using the previous example, the charge is prorated and applied on January 10 for the interval from January 10 to February 1. For subsequent cycles, the charge is applied on the billing date (from February 1 to March 1, March 1 to April 1, and so on).

About Prorating Recurring Charges and Rollovers

By default, PDC prorates any recurring charges and rollover amounts when customers do the following during a billing cycle:

Copyright © 2017, 2026, Oracle and/or its affiliates.

- Purchase charge offers
- Cancel charge offers
- Change their billing day of the month (DOM)

For example, suppose a customer cancels a subscription 15 days into a 30-day billing cycle. In that case, the customer is charged half of the cycle fee and receives half of the available rollover balance from the previous month.

However, you can configure PDC to apply all or none of the charge or rollover amount. For example, assume a customer's billing DOM is the 15th and on November 30 purchases a subscription with a $30 monthly fee. If you configure PDC to charge the total amount, the $30 monthly charge is applied to the customer's December 15 bill. If you configure PDC to charge nothing, no monthly charge is applied to the customer's December 15 bill.

You configure whether to prorate recurring charges and rollovers by using the following:

- The PDC UI. See "Adding a New Charge" in *PDC Online Help*.
- The **ImportExportPricing** utility. To do so, set the **<prorateFirst>**, **<prorateLast>**, and **<prorateCycle>** elements in your import XML file. For more information, see "Prorating Recurring Fees and Rollovers".

About Creating Charge Offers with Over 500 Charges

While creating large charge offers with 500 or more charges, PDC may generate the following error:

```
java.sql.SQLException: ORA-12899: value too large for column
```

To fix this, first try extending the VARCHAR2 data type to MAX_STRING_SIZE in your Oracle database. For more information, see "1.203 MAX_STRING_SIZE" in *Oracle Database Reference*.

> **Note**
>
> If the size of your charge offer data exceeds MAX_STRING_SIZE, check with Oracle Support before completing this step.

If you still receive an error after extending the VARCHAR2 data type, do this:

1. Go to the *BRM_Integration_Pack_home*/**PDC_BRM/sqlscripts/custom_scripts** directory, where *BRM_Integration_Pack_home* is the directory in which BRM Integration Pack is installed.
2. Do one of the following depending on the exception you receive:
   - For the following exception:

     ```
     SEVERE: Error inserting data in crossref table for BRE rateplan
     java.sql.SQLException: ORA-12899:
     value too large for column "XREF_BRE_RATEPLAN"."COMPOSITE_PM_IDS"
     (actual: 22015, maximum: 4000)
     ```


Copyright © 2017, 2026, Oracle and/or its affiliates.

In SQL*Plus, connect to your PDC database as the Xref user and then run the **UpdateXrefBreCompositePMIDsCol.sql** SQL script:

```
sqlplus XrefUser@database_alias
Enter password: password

SQL> @UpdateXrefBreCompositePMIDsCol.sql
```

where:

- *XrefUser* is the login name for the Xref schema user.
- *database_alias* is the PDC database alias.
- *password* is the password for *XrefUser*.

- For the following exception:

```
SEVERE: Error inserting data in pdc table java.sql.SQLException:
ORA-12899: value too large for column
"BCSIMPLEATTRIBUTEITEM"."PERMITTEDVALUES" (actual: 22015, maximum: 4000)
```

In SQL*Plus, connect to your PDC database as the PDC user and then run the **UpdateSimpleAttrItemPermittedValuesCol.sql** SQL script:

```
sqlplus pdcUser@database_alias
Enter password: password

SQL> @UpdateSimpleAttrItemPermittedValuesCol.sql
```

where:

- *pdcUser* is the login name for the PDC schema user.
- *database_alias* is the PDC database alias.
- *password* is the password for *pdcUser*.

3. Enter **exit** to quit SQL*Plus.
4. If the script runs successfully, restart the WebLogic Server, real-time rating engine (RRE), and batch rating engine (BRE).

Copyright © 2017, 2026, Oracle and/or its affiliates.

10

Configuring Pricing in Charge Offers

Learn how to configure pricing in Oracle Communications Billing and Revenue Management (BRM) charge offers.

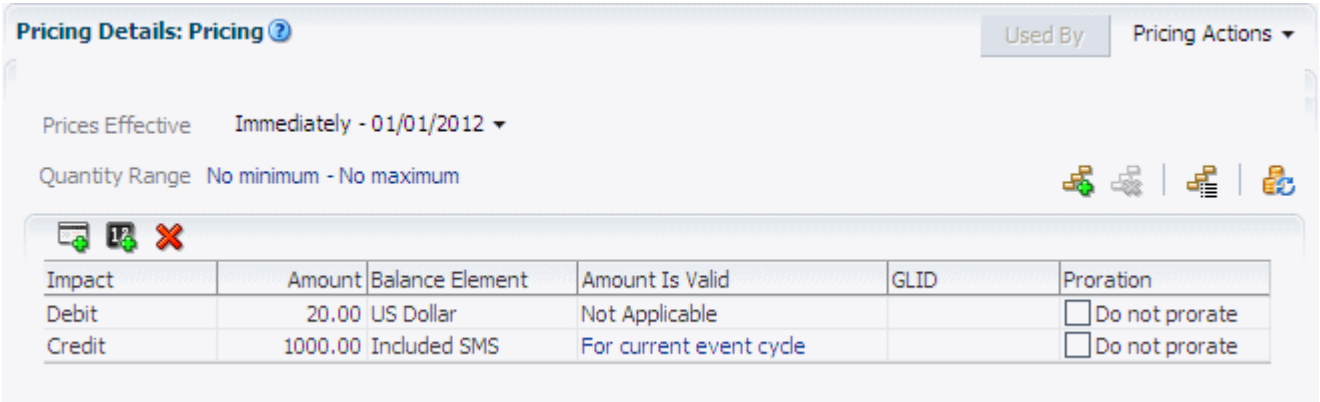
Topics in this document:

- Configuring Pricing in Charges
- Configuring Validity Periods for Noncurrency Credit Balances
- Granting Noncurrency Balances in Increments
- Specifying Rounding in Balance Impacts
- Specifying if a Balance Impact is Discountable
- About Minimum Charges
- About Conditional Balance Impacts
- About Prorating Charges
- Configuring Dynamic Charging

Configuring Pricing in Charges

Each charge can contain one or more balance impacts. You configure balance impacts in the pricing table. Figure 10-1 shows an example of the pricing table. Each row in the table represents a balance impact.

Figure 10-1 Pricing Table for a Charge



A balance impact is defined by how much to charge (the **Amount** column in the balance impact) for a given amount (the **Per Unit** column in the balance impact). When configuring pricing for a charge, you can also do the following:

- Configure when to grant noncurrency balances. See "Configuring Validity Periods for Noncurrency Credit Balances".

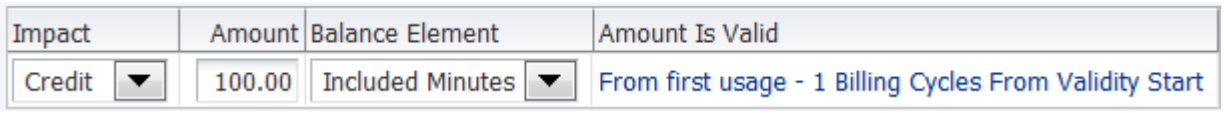
Copyright © 2017, 2026, Oracle and/or its affiliates.

- Configure how to round balance impacts. See "Specifying Rounding in Balance Impacts".
- Specify if a balance impact can be discounted. See "Specifying if a Balance Impact is Discountable".
- Specify a minimum amount to charge. See "About Minimum Charges".
- Disable proration for a charge if proration is enabled for a charge offer. See "About Prorating Charges".
- Use a price selector to choose a pricing configuration based on the values of specified event, service, or account attributes. See "About Selectors".

Configuring Validity Periods for Noncurrency Credit Balances

When you create an offer that credits noncurrency balances, such as granting minutes, you can configure a validity period for the balance impact. This is useful if you want to limit how long a customer can keep minutes in their balance before they need to purchase more minutes. Figure 10-2 shows a balance impact that grants 100 minutes that are available for one billing cycle.

Figure 10-2 Balance Impact for Minutes



You can specify the following fixed validity periods:

- For the current accounting cycle
- For the current billing cycle
- For the current recurring cycle

You can also define validity period start and end times. You can start a validity period at these times:

- When the event occurs.
- At first usage of the service.

  If a balance is shared among accounts in a sharing group, the balance validity period is set when any account in the group first impacts the balance. Because the same balance is shared with all accounts in the group, the validity period of that balance applies to all accounts.
- From a specified date.
- Relative to a specified date.

You can end a validity period based on:

- A date relative to the start date (measured in minutes, hours, days, months, or in account, billing, or recurring cycles).
- On a specified date.
- No end date.

If a charge offer or discount offer is canceled after being used, the validity end time of any balance granted by the offer is set to the time of the cancellation.

Copyright © 2017, 2026, Oracle and/or its affiliates.

> **Note**
>
> A charge offer that is valid from first usage should not grant a credit balance, such as included minutes, that is also valid from first usage. If it does, the granted balance cannot be consumed by the first usage event. See "About Granting First-Usage Balance Impacts In First-Usage Charges" for more information.

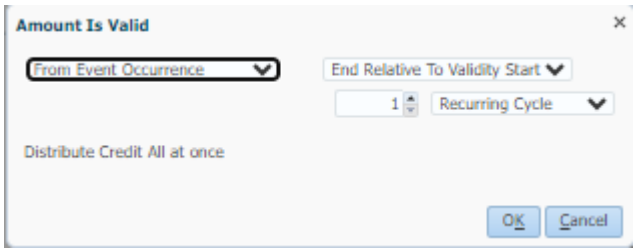
Example Configuration for Setting Validity Periods for Noncurrency Balances Based on First Usage Date

This example shows how to use PDC to configure the validity period of a charge to start at the first usage event's occurrence and end one cycle later.

In this example, the customer is using this feature and a customer's monthly billing cycle runs from October 10 to November 9 and the customer receives 500 free minutes, per cycle. If the first usage event is on October 15, the free minutes will expire on November 15. This means the free minutes remain valid for one billing cycle from the first usage event.

To configure the PDC UI for this example, set the field on the left to **From Event Occurrence**, and the fields on the right to **End Relative To Validity Start** and **1 Recurring Cycle**, as shown in Figure 10-3.

Figure 10-3 Amount Is Valid Dialog Box



See "Amount Is Valid Dialog Box" in *PDC Online Help* for more information about using this dialog box.

To configure this example using the **ImportExportPricing** utility, see "Specifying Validity Period to Start from Event Occurrence Date".

Granting Noncurrency Balances in Increments

> **Note**
>
> This functionality is supported only for recurring and one-time charges that have validity periods with a specified end date.

When you credit noncurrency balances, such as free minutes, in a one-time allotment or in a recurring cycle, you can choose to grant the noncurrency balance in smaller portions on an incremental basis, such as every 5 hours or every 2 days. For example, a one-time allotment

Copyright © 2017, 2026, Oracle and/or its affiliates.

of 50 Gigabytes that is valid for one month could be distributed to customers as one Gigabyte per day or 10 Gigabytes every seven days.

To configure PDC to grant a noncurrency balance in increments, you split the noncurrency balance into multiple validity periods and specify the following:

- The total validity period for the credit of noncurrency resources.
- The length of each incremental validity period, in terms of seconds, minutes, hours, days, or months.
- When the incremental bucket of noncurrency resources expire: at the end of the incremental validity period or at the end of the total validity period.

PDC calculates the amount of noncurrency resources allocated to each incremental bucket (except the last bucket) as follows:

```
(incremental validity period/total validity period) * (grant amount)
```

The amount for the last incremental bucket is calculated as follows:

```
(grant amount) - (total amount granted in previous buckets)
```

For example, assume your customers are granted a one-time allotment of 400 free minutes with a one-month total validity period. The 400 free minutes are split into multiple 7-day validity periods, with each incremental bucket of free minutes expiring at the end of its 7-day incremental validity period. Table 10-1 shows how these 400 free minutes would be distributed to your customers in the month of June.

Table 10-1 Schedule for Distributing 400 Free Minutes in 7-Day Validity Periods

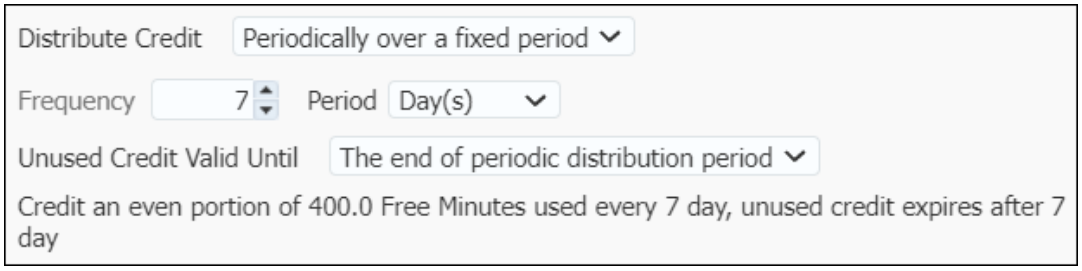
Bucket	Bucket Size Calculation	Bucket Size	Validity Start Date (starting at 12:00:00am)	Validity End Date (ending at 11:59:59pm)
1	(7/30) * 400	93 Free Minutes	June 1	June 7
2	(7/30) * 400	93 Free Minutes	June 8	June 14
3	(7/30) * 400	93 Free Minutes	June 15	June 21
4	(7/30) * 400	93 Free Minutes	June 22	June 28
5	400 - (93 * 4)	28 Free Minutes	June 29	June 30

You can configure multiple validity periods for noncurrency balances by using either the PDC UI or an external application.

- To configure multiple validity periods using the PDC UI, when adding balance impacts, set the **Distribute Credit**, **Frequency**, and **Unused Credit Valid Until** fields as shown in Figure 10-4. For more information, see "Adding Multiple Validity Periods to Noncurrency Balance Impacts" in *PDC Online Help*.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Figure 10-4 PDC UI for Configuring Multiple Validity Periods



- To configure multiple validity periods using an external application, call the PCM_OP_PRICE_SET_PRICE_LIST opcode and set the opcode's PIN_FLD_SPLIT_DETAILS input flist parameter. See "Committing Price List Data to the BRM Database" in *BRM Setting Up Pipeline Pricing*.

Specifying Rounding in Balance Impacts

In a balance impact, you can specify the number of units to which the charge applies. For example, for a mobile phone service, you might charge 0.40 US dollars for every minute of usage in 2-minute increments. In this case, if the unit is Minute, the Increment is 2.

If you specify an increment, you can select a rounding option. Rounding is based on the specified increment. For example, if a 3-minute-and-50-second call is rated in 2-minute increments, it is handled as follows:

- No Rounding: Not applicable. (Applies only to an increment of 1).
- Round Down: Charge for 2 minutes ($0.80).
- Round Up: Charge for 4 minutes ($1.60).

> **Note**
>
> BRM always prioritizes the **<incrementQuantity>** at the charge level over the **<IncrementStep>** at the pricing level. If both are configured in a charge using the PDC UI or the **ImportExportPricing** utility XML file, BRM selects the increment quantity.

Specifying if a Balance Impact is Discountable

When creating a balance impact, you can specify if it is discountable in a bundle. By default, balance impacts are discountable. When a charge type is discounted in a bundle, the discount applies to all instances of the charge type that are discountable. For example, you might have a charge offer that charges for two cycle events. You charge a monthly fee for one event, and you grant included minutes for the other event. If you specify a percentage discount for cycle charges in the bundle, the discount applies to both cycle charges. To discount only the monthly fee, you must make the balance impact for the monthly fee discountable and the balance impact that grants included minutes nondiscountable.

Copyright © 2017, 2026, Oracle and/or its affiliates.

About Minimum Charges

You can set a minimum charge for each balance element impacted by a pricing instance. Minimum charges are configured per price tier (see "About Defining Pricing Based on Multiple RUMs") and effective period.

If you define different minimum charges in different price tiers for the same balance element, the largest minimum charge is used during rating. For example, if one step defines a minimum charge of $2 and another a minimum charge of $1, $2 is used.

When multiple price tiers are applied to a charge, the minimum price is applied if the total of the charges does not meet the minimum charge. It is not applied for each of the price tiers if they each do not meet the minimum charge.

About Conditional Balance Impacts

A conditional balance impact adjusts a customer's balance by crediting or debiting it when they utilize a charge offer for the first time within a defined period, such as a day. For example, you can use conditional balance impacts to provide data to customers.

The starting point of the conditional balance impact period can be synchronized with either the activation of the associated charge offer, the beginning of a calendar day, or the occurrence of an event. You can also specify whether the balance is available from the period's start time or from the moment the event takes place.

Conditional balance impacts can only be used with charge offers associated with a service, not with the charge offers associated with an account.

The following example shows how to configure conditional balance impacts for granting included minutes. In this example:

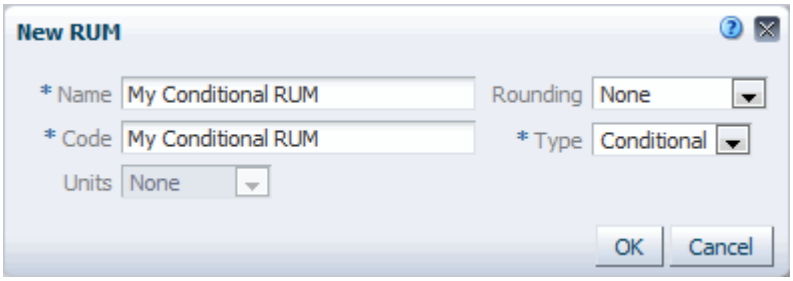
- A customer is offered 40 included minutes at $5 per day.
- The included minutes are valid for one day.
- The customer is not charged on days when the offer remains used.

To configure conditional balance impacts for granting daily included minutes:

1. Create a **Conditional** RUM for the specific service and usage event. See "Creating a RUM Configuration" in *PDC Online Help* for information.

   Figure 10-5 shows a RUM of type **Conditional**.

Figure 10-5 New Rum Configuration



2. Create a charge offer. See "Creating a Charge Offer" in *PDC Online Help*.

Copyright © 2017, 2026, Oracle and/or its affiliates.

3. Add a new charge to the charge offer, ensuring the following:
   - **Charge Category**: **Usage**
   - **Pricing Profile**: **Convergent Usage**
   - **Measured By**: The RUM created in step 1. You can select additional RUMs if needed.
   - The pricing name

   > **Note**
   >
   > The way ECE processes conditional balance impacts is dependent on the pricing name assigned to each charge in an offering. See "About Processing Conditional Balance Impacts" in *ECE Implementing Charging*.

4. Configure conditional balance impacts to grant 40 minutes at $5 per day on the days the charge offer is used.

   Figure 10-6 shows the conditions configured for granting daily included minutes.

Figure 10-6 Conditions for Granting Daily Included Minutes

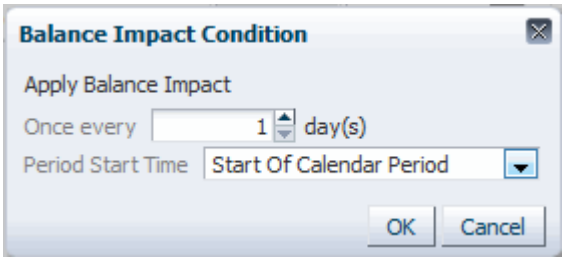


Figure 10-7 shows the validity period for the conditional balance impact.

Figure 10-7 Validity Period for a Conditional Balance Impact

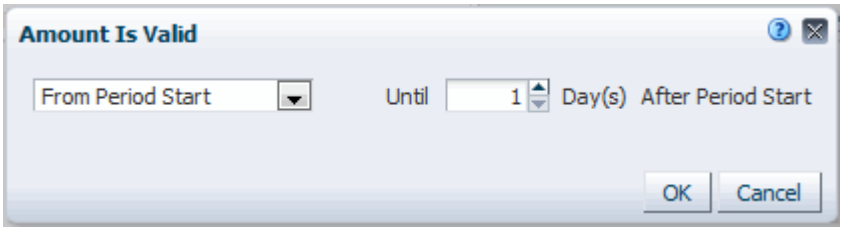
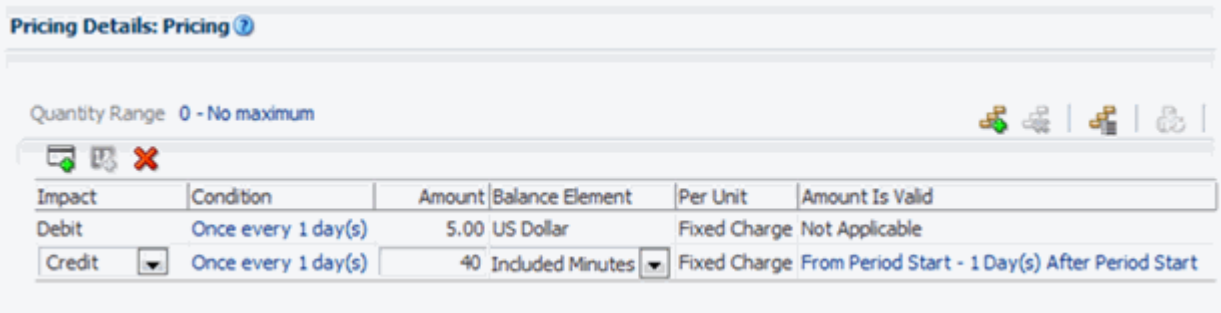


Figure 10-8 shows the conditional balance impacts used in this example for granting daily included minutes.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Figure 10-8 Conditional Balance Impacts for Granting Daily Included Minutes



About Prorating Charges

You can specify to not prorate a recurring charge when proration is enabled in the charge offer itself. The proration settings are specified in the charge offer; this option only enables or disables proration for each recurring charge.

Configuring Dynamic Charging

You can configure PDC to apply dynamic charging for one-time, recurring, and usage events by using price tags. Dynamic charging lets you specify different prices at purchase time according to rules defined in setup components.

You add price tags to balance elements of charges. For usage charges using the Convergent Usage pricing profile, you can also add price tags to quantity ranges.

> **Note**
>
> You cannot use price tags with chargeshare offers.

To configure dynamic charging:

1. Use the PDC UI or the **ImportExportPricing** utility to create price tag setup components. Specify the price tag name and permitted values, as well as balance elements, resource units, and services that the price tag can be associated with.

   For example, you could create a price tag called **FreeMinutesTag** and specify a range of valid values from 10 to 50. You can determine that it will be associated with the Free Minutes balance element, the minute resource unit, and the TelcoGSM and Telephony services.

   BRM stores the pricing tag configuration in the **/config/price_tags** object.

   See "Configuring Price Tags" for more information about creating price tag setup components.

2. Use one of the following to add the price tag, such as **FreeMinutesTag**, to balance elements or quantity ranges:

   - The PDC UI: Click the Price Tag icon in pricing table toolbar and then select a price tag from the Price Tag dialog box. For more information, see "Adding Pricing Tags for One-Time and Recurring Charges" or "Adding Price Tags for Usage Charges" in *PDC Online Help*.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- The **ImportExportPricing** utility: Use the **<priceTag>** element under the charge. See "Applying Price Tags to Charges".

When you add a price tag to a balance element, the rules configured in the setup component determine which price tags are available. For balance elements in charge offers, charge selectors, and price model selectors, price tags that match the balance element code, per unit value, and service associated with the charge are available.

When you add a price tag to a quantity range, the service names specified in the setup component determine which price tags are available. Other setup component fields aren't used to match price tags to usage events with quantity ranges.

3. At purchase time, enter the actual value to use in rating for the price tag. The rules configured in the setup component determine which values are valid. For example, if the setup component is configured with the **LIST** rule type, only the values configured in the list in the setup component are valid.

   BRM stores the price tag values in **/offering_override_values** objects.

   You can use BRM opcodes to get the list of valid price tags for a charge. See "Getting Information about Price Tags" in *BRM Opcode Guide*.

For information about enabling ECE to override the default value of the pricing attributes at run time, see "Pre-Rating Extension - Overriding Price in Product Offerings" in *ECE Implementing Charging*.

Copyright © 2017, 2026, Oracle and/or its affiliates.

11

Pricing Configurations in Charge Offers

Learn about various pricing configurations that you can use in Oracle Communications Billing and Revenue Management (BRM) charge offers using Pricing Design Center (PDC).

Topics in this document:

- Configuring Pricing Based on Quantity Ranges
- Configuring Pricing to Consume Granted Allowances Before Charging
- Configuring Effective Dates for Pricing
- About Defining Pricing Based on Multiple RUMs
- About Price Overrides
- About Fold Charge Pricing
- Granting First Usage Balances During Rating
- Automatically Renewing Granted Allowances

See also "Configuring Charge Offers" and "Configuring Charges in Charge Offers".

Configuring Pricing Based on Quantity Ranges

To calculate a charge based on the amount of usage or frequency of occurrence, you can create quantity ranges in charge pricing. For example, pricing for a telephony service might contain the following quantity ranges based on the total duration of calls during a month:

- No Minimum to 500 minutes: 10 cents per minute
- 500 to 1000 minutes: 5 cents per minute
- 1000 minutes to No Maximum: 1 cent per minute

> **Important**
>
> If you enable Oracle Communications Elastic Charging Engine (ECE) to generate midsession-rated events, note that each midsession event marks the end of a subsession of the network session. For the subsequent subsession, the network session's duration and volume counters are reset to 0. Therefore, if you use midsession-rated events, do not base your pricing on cumulative duration or volume across an entire network session.

To define pricing based on quantity, you do the following:

- Add quantity ranges and configure different balance impacts for each range. For example:
  - 0 - 10 minutes: 10 cents per minute
  - 10 - 60 minutes: 5 cents per minute
  - 60 minutes and above: 2 cents per minute

Copyright © 2017, 2026, Oracle and/or its affiliates.

- Specify how to apply the charge to the quantity ranges, which can be one of the following:
  - **Pick the quantity range containing the value:** ECE selects a single quantity range that contains the value of the quantity range expression. For example, if the customer uses 70 minutes, ECE charges $1.40 (70 minutes x $0.02 per minute).
  - **Distribute value across applicable quantity ranges:** ECE selects one or more quantity ranges that overlap the value of the quantity range expression. For example, if the customer uses 70 minutes, ECE charges $3.70 [(10 minutes x $0.10) + (50 minutes x $0.05 per minute) + (10 minutes x $0.02 per minute)].

When setting quantity range properties, you can also specify what happens when customers reach or exceed their credit limit for the charge. See "Allowing Customers to Exceed Their Credit Limit".

When you use quantity ranges in combination with time ranges, you could specify if the quantity consumed is relevant to a new time range. See "Rating Events Split Across Time Periods" for information.

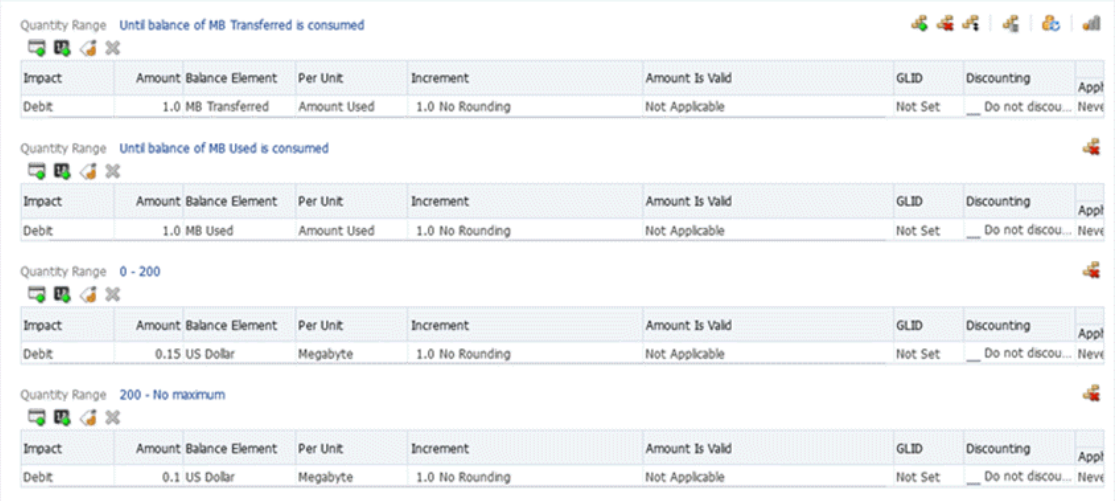
Configuring Pricing to Consume Granted Allowances Before Charging

You can create usage charges that consume a customer's granted allowances before charging for any usage. For example, you could create a usage charge that first consumes a customer's monthly allowance of free GBs of data and then charges $10 for any subsequent GBs used. You implement this functionality by adding quantity ranges to usage charges.

> **Note**
>
> Consuming granted allowances before charging for usage fees is supported by the ECE rating engine only.

Figure 11-1 shows how to create sample quantity ranges in a usage charge using the PDC UI.

Figure 11-1 Sample Pricing for Consuming Free MB Before Charging



Copyright © 2017, 2026, Oracle and/or its affiliates.

This example specifies to first consume the entire balance of transferred MBs, then consume the entire balance of MB used, then charge 15 cents per MB for the next 200 MBs, and then charge 10 cents per MB for any subsequent MBs.

Using this example, assume a customer has 200 transferred MBs and 250 used MBs in a month. If the customer consumes 800 MBs of data in that month, ECE would deplete the 200 transferred MBs, deplete the 250 used MBs, charge $30 for 200 MBs, and charge $10 for the remaining 150 MBs.

You configure usage charges to consume granted allowances before applying charges by using:

- The PDC UI. To do so, use the Add Quantity Range dialog box. See "Adding Ranges for Usage Pricing" in *PDC Online Help*.
- The **ImportExportPricing** utility. To do so, in your input XML file, set the **<upperBoundExpression>** element under the **<priceTierRange>** element. See "Consuming Granted Allowances Before Applying Usage Charges".

> **Note**
>
> Allowances are consumed before applying standard usage charges. In configurations with both Discount Sharing Groups (DSG) and shared allowances, PDC first charges for usage that exceeds the member's allowance. Then, it applies the DSG discount based on the remaining usage. For more information on DSG discounts, see "About Discount Sharing" .

Configuring Effective Dates for Pricing

You can apply a date range to a pricing configuration; for example, set a fixed start date and an end date. You can include multiple date ranges in a pricing configuration. This enables you to implement future price changes for a charge without creating multiple versions of the charge. For example, you could create a charge that has a $10 recurring charge valid for one month, and $20 after that.

Date ranges cannot overlap, for example:

- Immediately through 6/1/2015
- 6/1/2015 through 1/1/2016
- 1/1/2016 through never ends

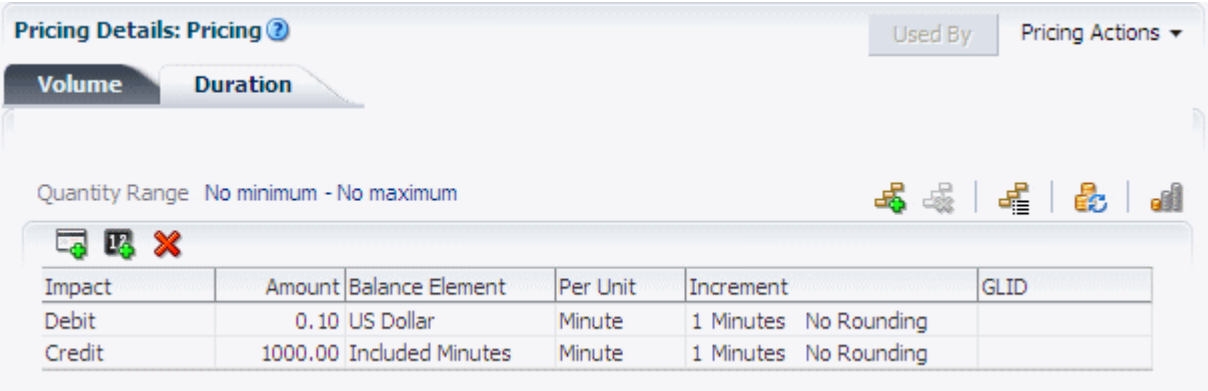
When configuring the charge offer, you can also configure whether customers who subscribe during one date range will keep the same pricing configuration after that range ends, or move to the pricing configuration on the next date range, and whether the pricing configuration is chosen based on the service instantiation date or the purchase date. See "Configuring How Pricing Configurations for Date Ranges are Applied" for more information about how the pricing configurations are determined.

About Defining Pricing Based on Multiple RUMs

To configure pricing for multiple RUMs within a charge, you use multiple price tiers, one for each RUM. Figure 11-2 shows a charge that contains two price tiers, Volume and Duration.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Figure 11-2 Volume and Duration Price Tiers



About Price Overrides

A price override enables you to replace the balance impact of a record imported from another system or to adjust the balance impact by a specified percentage or a fixed amount. For example, you might charge for content downloads that have already been rated by another service provider. A price override enables you to charge a different amount.

About Fold Charge Pricing

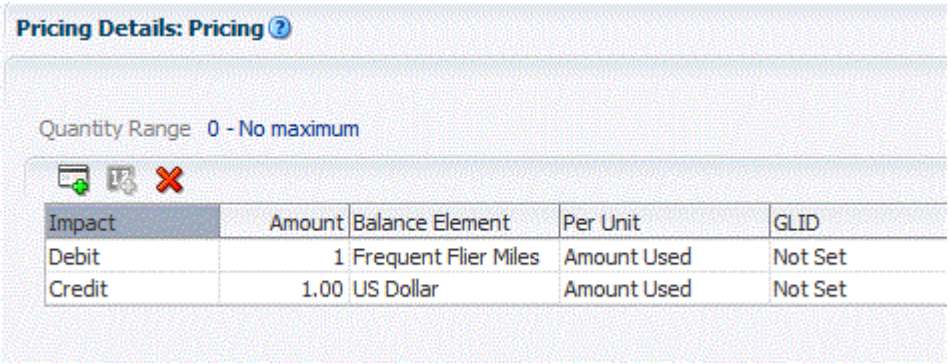
A fold charge is used to zero-out a balance, such as unused minutes or to convert one balance to another.

A fold is configured using a charge selector. The event for the charge selector must be the **Fold** event. After you have configured the charge selector, select it in the charge offer.

The pricing in a fold charge should have a debit for the balance that you are converting and one or more credits for the balances that you are converting to.

Figure 11-3 shows the pricing for a fold that converts Frequent Flyer Miles balance to US Dollar balance, $1 for each frequent flyer mile.

Figure 11-3 Fold Pricing to Convert One Balance to Another



See "About Folds" for information about how fold balances are managed.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Granting First Usage Balances During Rating

ECE can grant first-usage balance impacts during rating. For example, you could create a charge offer that includes these balance impacts:

- Debit 5 cents per minute if there are no included minutes.
- Credit 1 minute for each minute paid at 5 cents per minute. These minutes are valid on first usage.

In this example, the charges occur as follows:

1. A subscriber has used up all their included minutes and is being charged 5 cents per minute.
2. After ten minutes, the subscriber terminates the call. The subscriber is granted 10 minutes.
3. The next call that the subscriber makes uses the ten minutes, granted as first-usage balance impacts.

You can also grant first-usage balance impacts by using a discount. For example, you could create:

- A charge offer that charges 5 cents per minute.
- A discount that credits one SMS message for each called minute.

In this example, the charges occur as follows:

1. A subscriber makes a 10-minute call.
2. The subscriber terminates the call. The subscriber is granted 10 SMS messages, valid at first usage, with a validity end date after 30 days.

Automatically Renewing Granted Allowances

You can configure granted allowances to automatically renew once the resources of the previous grant have been depleted. This provides your customers continuous access to the granted resource without requiring manual renewal.

> **Note**
>
> - When ECE processes granted allowances, it handles grant values as LONG values instead of INTEGER values. This enables you to configure large granted allowance values without causing integer overflow during ECE processing.
> - Automatic renewals of granted allowances are currently supported only for usage events that are processed by the ECE rating engine

You can configure granted allowances to renew automatically by using:

- The PDC UI. To do so, see "Automatically Renewing Granted Allowances" in *PDC Online Help*.
- The **ImportExportPricing** utility. To do so, in your input XML file, use the **<AutoRenewResourceExpression>** element. See "Configuring the Automatic Renewal of Allowances".

Copyright © 2017, 2026, Oracle and/or its affiliates.

12

Creating Discount Offers

Learn how to create discount offers in Pricing Design Center (PDC) for Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

- About Discount Offers and Discounts
- Configuring How Discount Offers are Purchased and Applied
- Applying Discounts Based on Quantity
- About Discount Filters
- About Discount Triggers
- About Using Temporary Balances in Discounts
- Using Functions to Calculate a Balance Impact
- Applying Multiple Discount Versions to an Event
- About Billing-Time Discounts
- Configuring Effective Dates for Discounts
- Configuring Discount Validity
- Discount Validity Rounding
- Discounts and Tax Calculation
- Discounts and General Ledger
- About Snowball Discounts
- Applying Price Tags to Discount Offers
- Tying Discount Offers with Charge Offers

About Discount Offers and Discounts

A discount offer is a purchasable product offering, similar to a charge offer. You can use discount offers to do the following:

- Reduce a charge by a percentage. For example, reduce the monthly fee by 50%.
- Track usage or spending by using counters and discount the charge for an event based on the value of the counter. For example, a discount can track usage and apply a 25% discount to the charge of an event when over 1000 minutes have been used.
- Consume or grant balance impacts. For example, if an event is charged one dollar per minute, you can use a discount offer to allow the customer to use their included minutes and credit the corresponding charge. The discount offer would include a discount with two balance impacts:
  - Consume one minute for every dollar charged.
  - Credit one dollar for every minute consumed.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- Consume balance impacts immediately instead of reserving them. You can do this using the **Instant Debit** impact type. For example, if an Instant Debit of 2 dollars is fixed for a usage event, 2 dollars would be added to the bucket every time a usage event is triggered.

> **Note**
>
> The Instant Debit impact type is supported only on ECE-enabled systems and is applicable to currency resource in a usage event.

A discount offer includes one or more discounts. Every discount has one or more balance impacts. The balance is represented by a currency or noncurrency balance element. A debit balance must be a noncurrency balance element, such as minutes.

Figure 12-1 shows a balance impact for a 10 percent discount on a monthly recurring charge.

Figure 12-1 Discount Reducing a Charge by a Percentage

Impact	Balance Element	Amount	Per Unit	What to Discount
Credit	US Dollar	10.00	Percent of	Charge

A discount is calculated based on two elements in the balance impact, the values in the **Per Unit** column and the **What to Discount** column. The **Per Unit** value is how to apply the balance element amount to the **What to Discount** value.

In Figure 12-1, the **Per Unit** value is a percentage, applied to the charge (the **What to Discount** value). Because the event is charged in US Dollars, the balance impact is also in US Dollars. A currency balance element must be the same balance element in the charge that the discount is granted to.

The **Per Unit** value can be a percentage, an absolute value, or incremental; for example, apply the discount for every 10 gigabytes downloaded. The **What to Discount** value can be based on a charge, a quantity, a balance, or a calculation of multiple values.

Figure 12-2 shows a discount that grants 5 minutes for every 10 gigabytes of data downloaded. The **What to Discount** value in this case is the value that is used to calculate the discount by applying the **Per Unit** value.

Figure 12-2 Grant 5 Minutes for Every 10 Gigabytes

Impact	Balance Element	Amount	Per Unit	What to Discount
Credit	Included Minutes	5.00	For every 10 of	Balance[Gigabytes Counter]

Figure 12-3 shows a discount that credits $5 on a purchase fee. In this case, the **Per Unit** value is Fixed, so the **What to Discount** value is not needed for calculating the discount and is set to **Not Applicable**.

Figure 12-3 $5 Discount on a Purchase Fee

Impact	Balance Element	Amount	Per Unit	What to Discount
Credit	US Dollar	5.00	Fixed	Not Applicable

Copyright © 2017, 2026, Oracle and/or its affiliates.

A discount offer applies to a single service. You can also configure a discount offer to apply to the account. In that case, a discount would apply to charges for the same event for any service owned by the account. Discount offers that apply to a service are not valid when the service is inactive or canceled. Account discount offers are valid even if no service is active.

You can also make a discount offer eligible for tying with specific charge offers at purchase time. When a discount offer is tied to a charge offer, BRM applies the discount only to the purchased charge offer or charge offers that share the same tie ID. This lets you target a discount to a specific charge in a bundle, instead of applying it to every eligible charge for the service and event type. See "Tying Discount Offers with Charge Offers".

Within each discount offer, each discount has a discount type based on the events configured for the service that the discount offer applies to. Discount types are organized in discount categories as follows:

- **Usage:** These discount types discount usage events. For example, you might use the following:
  - The ConvergentUsage discount type to discount Wireless Voice events.
  - The ConvergentData discount type to discount Wireless Data events.
- **Recurring:** These discount types discount recurring events, such as a monthly fee. For example, recurring discount types might be Monthly Recurring Event and Annual Recurring Event.
- **One-time:** These discount types discount one-time events, such as purchase events or cancel events.
- **Billing Time:** This discount type is calculated when billing is run, instead of when the charge is rated. See "About Billing-Time Discounts" for information.

For recurring discounts, note the following:

- Although a charge offer cannot contain both cycle forward and cycle arrears charges, a discount offer can contain both types of cycle discounts as long as their frequency (monthly, yearly, and so on) is the same.
- A recurring discount is applied to all the recurring charges for the same service. For example, if there are different Monthly Recurring Event charges for the same service, the Monthly Recurring Event discount applies to all of them. You can use discount filters to apply different discounts to them, by selecting different pricing names. See "About Discount Filters".

When you create a discount offer in PDC, you configure settings that apply to the offer, for example, when the discount offer is available for purchase. See "Configuring How Discount Offers are Purchased and Applied".

When you create a discount, you can define the following:

- You can calculate discounts based on quantity, for example, the quantity of minutes used. See "Applying Discounts Based on Quantity".
- You can use filters to specify criteria that a charge must satisfy to be eligible for the discount. See "About Discount Filters".
- You can use discount triggers to restrict the application of a discount until certain conditions are met. See "About Discount Triggers".
- You can use the results of one discount to calculate another discount for the same event. See "About Using Temporary Balances in Discounts".
- You can calculate discounts based on data in the BRM database, or other data that is not directly configurable in PDC. See "Using Functions to Calculate a Balance Impact".

Copyright © 2017, 2026, Oracle and/or its affiliates.

- You can use rules to apply multiple filters, triggers, and pricing to the same event. See "Applying Multiple Discount Versions to an Event".
- You can use counters to create a value that can provide a discount when billing is run. See "About Billing-Time Discounts".
- You can use date ranges to apply different discounts based on a fixed date range. See "Configuring Effective Dates for Discounts".
- You can configure how a discount is applied when it is valid for only part of a cycle. See "Configuring Discount Validity".
- You can create a billing-time discount that distributes a percentage discount to all accounts in a discount sharing group. See "About Snowball Discounts".
- If you configured product specification attributes for discount offers in XML templates, you can set their values. See "Configuring Product Specification Attributes for Pricing Components".
- If you configured price tag setup components, you can apply price tags to discount offers at the balance element level. See "Applying Price Tags to Discount Offers".

Configuring How Discount Offers are Purchased and Applied

You can control how a discount offer is purchased and applied by configuring the following:

- Subscription discount offers and system discount offers:
  - **Subscription:** The discount offer applies only to the customer who owns the discount offer.
  - **System:** The discount offer applies to all customers who use the specified service. Use system discounts when you do not want to re-create the same discount multiple times in your product offerings.
- A range of dates during which the discount offer can be purchased. To be added to a bundle, a discount offer must have a purchase period that is the same as or greater than the bundle's purchase period.
  You can also specify the effective period for the discount offer when you configure the bundle that the discount offer belongs to. For example, you can set the discount to be effective one month after it is purchased. See "About the Validity Periods of Offers in Bundles".
- How many instances of the discount offer that the customer can purchase and own.
- Whether the discount offer can be automatically shared with all members of a billing hierarchy.
- Whether the discount offer can be tied to specific charge offers at purchase time. Only discount offers marked as tied can be associated with specific purchased charge offers. See "Tying Discount Offers with Charge Offers".
- The priority of each discount offer. Offers with higher priority numbers are applied before those with lower priority numbers.
- How to apply multiple discount offers to a single charge. See "Applying Multiple Discount Offers to an Event" for information.
- How to configure mutually exclusive discount offers. See "Configuring Mutually Exclusive Discount Offers" for information.
- What happens when a customer purchases the same subscription discount offer multiple times. The discount offer can be purchased as a new subscription or a replacement of the existing subscription. See "Purchasing the Same Offer in a Bundle Multiple Times".

Copyright © 2017, 2026, Oracle and/or its affiliates.

Applying Multiple Discount Offers to an Event

The same charge or portion of a charge can be eligible for multiple discount offers. When this occurs, BRM uses the discount offer priority and the following options to determine how to apply the discount offers to the charge:

- **Original Charge:** The discount offer is applied to the original charge amount, regardless of whether that amount was reduced by previous discount offers.
- **Remaining Charge:** The discount offer is applied to whatever charge amount remains after previous discount offers are applied.
- **Remaining Charge and Quantity:** The discount offer is applied only to the part of the charge and quantity that was not discounted already by a previous discount offer. For this option, the discount offer can be used only if:
  - Part of the charge has not yet been evaluated for a discount.
  - The discount offer consumes noncurrency balances or reduces currency charges.
  - The discount does not grant balances.

For example, a bundle includes the following discount offers:

- First priority: 10% off
- Second priority: 20% off

The customer makes a 100-minute call at $0.10 a minute for a total charge of $10. The discount for the charge is calculated as follows:

- Using **Original Charge:**
  - First discount: 10% of $10 = $1 discount
  - Second discount: 20% of $10 (the original charge) = $2 discount
  - Total discount = $3
- Using **Remaining Charge:**
  - First discount: 10% of $10 = $1
  - Second discount: 20% of $9 (the remaining charge) = $1.80
  - Total discount = $2.80
- Using **Remaining Charge and Quantity:**
  - First discount: 10% of $10 = $1
  - Second discount: Not applied because there is no remaining charge that has not been discounted.
  - Total discount = $1

Configuring Mutually Exclusive Discount Offers

You configure discount offer exclusions to prevent a discount offer from being used or purchased when another discount offer or package is owned. You can use two methods to configure mutually exclusive discounts:

- Discount offer-to-discount offer exclusions. You can configure a discount offer to exclude the application of other discount offers when both discount offers apply to the same event. You configure discount offer-to-discount offer exclusions when you create discount offers.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- Discount offer-to-package exclusions. You can configure a package to exclude specified discount offers. You specify discount-to-package exclusions when you create packages.

  Discount offer-to-package exclusions can be configured to operate in two ways:

  - **At purchase.** If an exclusion relationship exists between a discount offer and a package, a customer can own the discount offer or the package, but not both. Further, the customer cannot own any discount offers associated with the package if the customer owns the excluded discount offer.
  - **At run time during charging.** In this case, the customer can purchase the excluded discount, but it is not applied during charging or billing.

When configuring discount exclusions, note the following:

- To exclude a billing-time discount, you must also exclude the counter discount associated with the billing-time discount. See "About Billing-Time Discounts" for more information.
- Discount exclusion takes precedence over discount priority. That is, a discount can be excluded even if it has a higher priority than the discount that is excluding it.
- System discounts can be excluded.
- Shared discounts can be excluded.

Discount exclusions are disabled by default. To enable discount exclusions, see "Enabling Mutually Exclusive Discount Offers" in *BRM Managing Customers*.

Applying Discounts Based on Quantity

To apply discounts based on usage amount or on an amount in a balance, you can create quantity ranges. For example, a discount for a mobile phone service might contain the following quantity ranges:

- 0 through 500 minutes: No discount
- 500 through 1000 minutes: 10% off
- 1000 minutes through No Maximum: 15% off

In this example, the quantity is based on how many minutes are used. You can also base the quantity on the charge, an amount in a balance, or a calculated value.

The start and end values of all quantity ranges must be decimal except that you can use a calculated value to specify the end value of the last quantity range. For example, to use all the minutes in a balance, the calculation would be:

0 - Balance[Included Minutes]

In addition to specifying what to base the quantity on (usage, charges, and so on), you specify how to apply the discount to the quantity ranges. There are two options:

- **Pick the quantity range containing the value:** Applies the discount in the range that contains the entire quantity amount. Using the previous example, if the value is 700, the ranges are applied as follows:
  - 0–500: Does not qualify.
  - 500–1000: The discount for this range is applied to the entire result.
  - 1000-Maximum: Does not qualify.

Copyright © 2017, 2026, Oracle and/or its affiliates.

> **Note**
>
> In this option, the start value for the quantity range is inclusive and the end value is exclusive. For example, a range of 0 through 500 applies to quantities greater than or equal to 0 and less than 500.

- **Distribute value across applicable quantity ranges:** The pricing in each range is applied to the amount of the basis that falls within it. If the value is 700, the ranges are applied as follows:
  - 0–500: The discount for this range is applied to the first 500 minutes.
  - 500–1000: The discount for this range is applied to minutes from 500 to 700.
  - 1000-Maximum: Does not qualify.

You can use absolute value and square root functions in quantity range expressions when setting discounts. Use these to calculate discount tiers based on the magnitude or the square root of the quantity.

When you distribute the value across ranges, you can use StepCharge or StepQuantity for the What to Discount value. This ensures that the discount is applied to only part of the quantity range:

- StepCharge bases the discount on the charge that applies to the quantity range, not the entire charge.
- StepQuantity bases the discount on the event quantity that applies to the quantity range.

About Discount Filters

A discount filter specifies criteria that a charge or a portion of a charge must satisfy to be eligible for the discount. For example, a discount might apply only to calls made on weekends or during December.

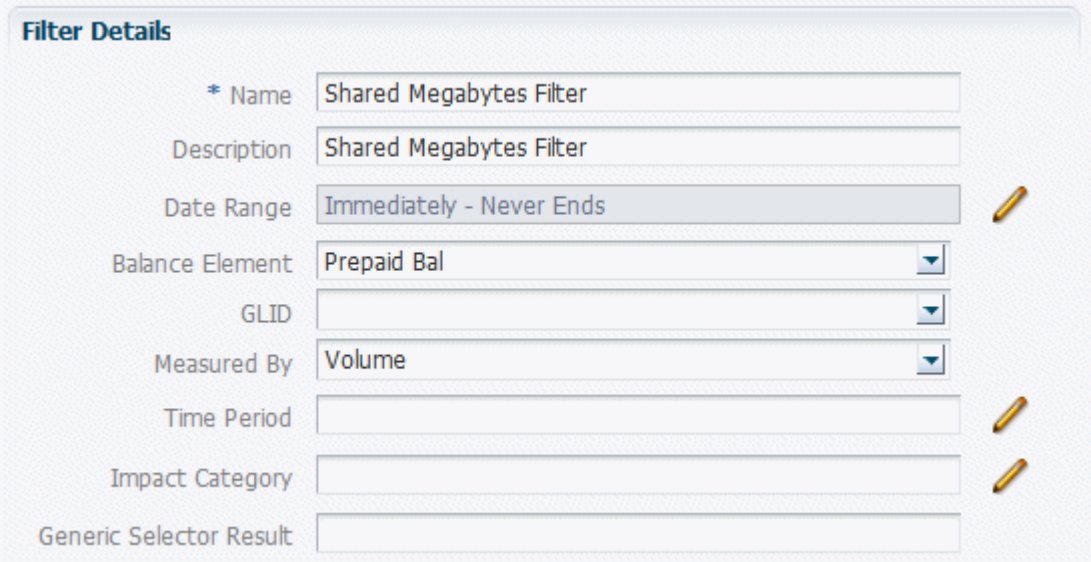
Filter criteria can be defined by specific values (such as a time period from a specified time model) or by using regular expressions.

You can create filters based on several values; for example, the amount consumed by the event, a balance element, a time period (peak and off-peak) and so on.

Figure 12-4 shows a discount filter that applies a discount to the event if it is measured in volume (not duration) and the balance element is a prepaid balance.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Figure 12-4 Discount Filter



When creating a discount filter:

- You can specify a date range that defines when the filter is valid.
- If a discount has multiple filters, all filters are applied.
- A filter can be reused by multiple discounts.
- You can enter NOT operator in the following fields to specify the values to be excluded:
  - **GLID**
  - **Time Period**
  - **Impact Category**
  - **Generic Selector Result**

> **Note**
>
> PDC does not validate the expressions entered in the fields.

Click the **Search** icon to search for existing filters and add to the discount.

About Discount Triggers

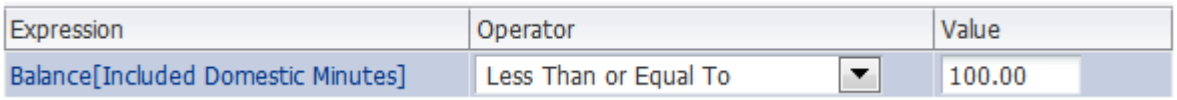
You can use discount triggers to restrict the application of a discount until certain conditions are met. For example:

- The number of domestic call minutes is less than or equal to 100.
- The bytes used in a GPRS session must be greater than 1 MB.

Figure 12-5 shows a discount trigger in PDC.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Figure 12-5 Discount Trigger



If a discount trigger includes multiple conditions, all conditions must be met for the discount to be granted.

Discount triggers accept absolute value and square root functions in their conditions. Use these functions in trigger expressions when you want to evaluate the magnitude or square root of a value before applying a discount.

A discount can have only one trigger. A trigger can be used by multiple discounts.

About Using Temporary Balances in Discounts

Typically, you use temporary balances when you need the results of one discount to calculate another discount for the same event.

For example, consider a discount that awards 5 text messages if the amount of data transferred in a data usage event exceeds 1 megabyte and the session duration exceeds 30 minutes. In this situation, Discount 1 would determine the amount of data sent or received and store it in Temporary Balance A. Discount 2 would determine the duration of the session and store it in Temporary Balance B. Discount 3 would grant 5 text messages if the data in Temporary Balance A exceeds 1 megabyte and the session length in Temporary Balance B exceeds 30 minutes.

You can only use noncurrency balances for temporary balances. Rounding rules do not apply to them.

Unlike other balances, temporary balances are maintained only while a single event is being discounted. If multiple discounts are applied to a single event, the temporary balance is maintained until all the discounts are processed.

For billing-time discounts, temporary balances are maintained only while a single discount is processed. Therefore, you cannot use temporary balances for billing-time discounts to pass a balance impact from one discount offer to another.

Using Functions to Calculate a Balance Impact

Use a function in a discount to calculate a value that is not directly configurable in PDC. The function typically performs a database query. For example, it could query the total charges for the ten most frequently called numbers. This amount can then be discounted.

When configuring a discount's balance impact, you can apply absolute value and square root functions to your calculations. This lets you adjust balance impacts based on the magnitude or square root of input values.

For example, suppose you want to implement a 10% discount on data usage from an access point. To implement this discount, you could include a function as the **What to Discount** value:

Function["AccessPoint"]

When BRM processes this discount, the usage amounts for the access point are substituted into the **What to Discount** value. A 10% discount is applied to AccessPoint usage.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Applying Multiple Discount Versions to an Event

You can apply multiple versions of a discount to the same event. To do so, you create multiple discount rules. For example, you might use discount rules to apply a discount in peak or off-peak times. A discount rule includes all of the elements of a discount: filters, triggers, and pricing. The rules are applied in the order that they appear in PDC.

You can use the same criteria (**Original Charge**, **Remaining Charge**, and **Remaining Charge and Quantity**) for applying multiple discount rules as you do when you apply different discount offers to the same event. See "Applying Multiple Discount Offers to an Event".

About Billing-Time Discounts

Billing-time discounts are calculated at the end of the billing cycle. This enables you to apply discounts based on the aggregation of a balance during a billing cycle. For example, you can create a billing-time discount to do the following:

- Reduce a balance by $10 if the total usage charges for the billing cycle are more than $100.
- Grant 10 text messages if the total minutes used during the billing cycle are more than 500.

Billing-time discounts are typically based on balances accumulated by using a counter. The counter can be used to apply usage, recurring, or one-time discounts.

For example, you can create a billing-time discount that grants 10 minutes for every $100 in usage charges. To create a billing-time discount, you set up the following discounts:

- A non-billing-time discount that increments a counter to track Dollars Spent. Every $1 charged increments the counter by 1.
- A billing-time discount grants 10 minutes if the Dollars Spent counter is over 100 at billing time.

Configuring Effective Dates for Discounts

You can add a date range to a discount by specifying a fixed start date and end date (discount date ranges cannot be relative). The date range is the period during which the discount is effective.

This enables you to use different versions of the same discount during different time periods by adding multiple date ranges to a single discount rather than by creating multiple discounts.

You can use different discount rules and pricing for each date range. For example, you could create a discount that applies a 10% discount for the first date range and a 20% discount for the second date range.

Date ranges cannot overlap in the same discount. For example:

- Immediately through 6/1/2016
- 6/1/2016 through 1/1/2017
- 1/1/2017 through never ends

Copyright © 2017, 2026, Oracle and/or its affiliates.

> **Note**
>
> Date ranges for different discounts in the same discount offer can overlap.

Configuring Discount Validity

You can configure discount validity for the following cases:

- Specifying Discount Behavior for Partial Cycles
- Applying a Discount After the Discount Offer Is Inactivated or Canceled
- Configuring Validity Periods for Noncurrency Credit Balance Impacts

Specifying Discount Behavior for Partial Cycles

You can set how to apply a discount when it is valid for only part of a cycle. For example, when a customer purchases, cancels, or inactivates a discount during an active cycle.

For each of these scenarios, specify whether the discount applies to the full cycle, a partial cycle, or does not apply:

- A discount starts mid-cycle and remains valid through the cycle end. Mid-cycle is any time after the cycle start and before the cycle end.
- A discount starts at the cycle start and ends mid-cycle.
- A discount starts mid-cycle and ends before the end of the cycle (that is, also mid-cycle).

> **Note**
>
> These validity rules apply to discount operations only. When a service is purchased, canceled, or inactivated during a cycle, BRM also performs the action on all associated products and applies discounts according to the product proration settings. You set the product proration settings when you add a charge to a charge offer. For more information, see "Cycle Settings Section for Recurring Charges" in *PDC Online Help*.

For each scenario, you specify whether the discount is applied for the full cycle, applied to part of the cycle, or is not applied at all.

For example, Figure 12-6 shows a discount that becomes valid in mid-cycle, but is applied to the entire cycle.

> **Note**
>
> In this example, if the discount is a usage discount, the charges must be rerated to apply the discount to the events that occurred before the discount was valid.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Figure 12-6 Valid Mid-cycle - Full Discount

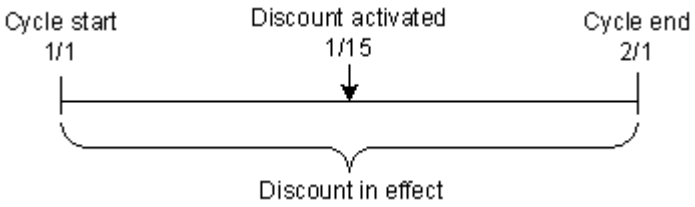


Figure 12-7 shows a discount that becomes valid in mid-cycle that is applied to only part of the cycle.

Figure 12-7 Valid Mid-cycle - Partial Discount

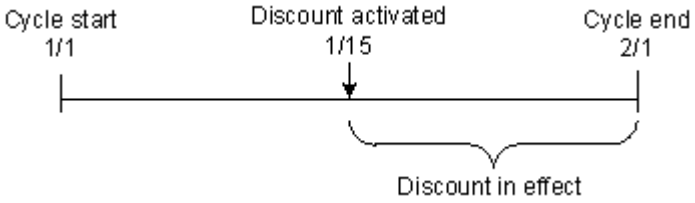
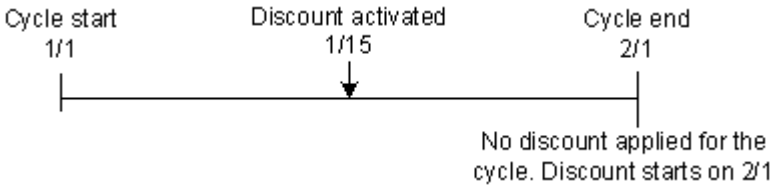


Figure 12-8 shows a discount that becomes valid in mid-cycle, but no discount is applied.

Figure 12-8 Valid Mid-cycle - No Discount



Applying a Discount After the Discount Offer Is Inactivated or Canceled

For each discount, you specify whether to apply the discount after the discount offer is canceled or inactivated. This controls how to apply discounts to events that occurred prior to the discount offer cancellation or inactivation. The options are:

- Don't stop discounting if the discount offer is inactive or canceled
- Stop discounting if the discount offer is inactivated
- Stop discounting if the discount offer is canceled
- Stop discounting if the discount offer is inactivated or canceled

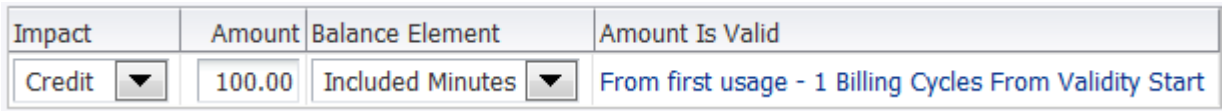
Configuring Validity Periods for Noncurrency Credit Balance Impacts

When you create a discount balance impact that credits noncurrency balances, such as granting minutes, you can configure a validity period for the balance impact. This is useful if you want to limit how long a customer can keep minutes in their balance before they need to

Copyright © 2017, 2026, Oracle and/or its affiliates.

purchase more minutes. Figure 12-9 shows a balance impact that grants 100 minutes that are available for one billing cycle.

Figure 12-9 Balance Impact for Minutes



You can specify a validity period corresponding to the current accounting cycle or the current billing cycle. You can also define validity period start and end times. You can start a validity period:

- When the event occurs.
- At first usage. Balances that start on first usage are added to the account balance at the time of the grant, but the validity starts when the subscriber uses the service for the first time.
- From a specified date.
- Relative to a specified date.

You can end a validity period based on:

- A date relative to the start date (measured in minutes, hours, days, months, or in account, billing, or recurring cycles)
- On a specified date.
- No end date.

If a charge offer or discount offer is canceled after being used, the validity end time of any balance granted by the offer is set to the time of the cancellation.

Discount Validity Rounding

When you create a discount offer, you can specify whether a discount offer's validity period starts when the discount is purchased or at midnight of the day it was purchased.

You can configure discount validity rounding using the following:

- The PDC UI. See "Offer Settings Subsection" in *PDC Online Help*.
- The **ImportExportPricing** utility. To do so, in your input XML file, set the <**validityRounding**> element under the <**alterationOffering**> element. See "Configuring Discount Validity Rounding".

If not configured, PDC uses the system-wide setting, which you configure using the **TimestampRounding** business parameter. See "Configuring Time-Stamp Rounding" in *BRM Configuring and Running Billing*.

Discounts and Tax Calculation

You can apply a tax code to a discount. You need to define tax codes before you can assign them in discounts. You can reference the tax code directly, or you can use a tax code selector.

For information, see "About Calculating Taxes" and "Creating Tax Codes" in *BRM Calculating Taxes*.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Discounts and General Ledger

You assign a general ledger ID (G/L ID) to discount balance impacts. This enables you to track revenue for each type of discount.

About Snowball Discounts

A snowball discount is a type of shared billing-time discount that distributes a percentage discount to all accounts in a discount sharing group. A discount sharing group allows one account (the group owner) to share discounts with other accounts (the group members).

To implement a snowball discount, you use two discounts:

- A discount that increments a counter.
- A billing-time discount that calculates and distributes the discount based on the counter balance. In PDC, you designate this as a snowball discount.

For example, a discount sharing group has one owner and three members. The owner purchases a discount offer that includes a snowball discount that grants $0.01 for every minute of telephone usage. At the end of the billing cycle, the group used 4,000 minutes. A total of $40 is granted to the group. If the discount is configured to divide the grant evenly, the owner and each member receive $10.

You can also configure the discount to distribute the grant based on the minutes used for each account. For example, if the owner uses 2,500 minutes, and each member uses 500 minutes, the owner receives a $25 grant, and each member gets a $5 grant.

In addition to creating the counter discount and the billing-time discount, you need to do the following:

- Set up a discount sharing group by defining the group owner, group members, and a list of the discounts that the group will share. See "About Discount Sharing" for more information.
- Specify how the snowball discount is distributed in the **pin_snowball_distribution** file and run the **load_pin_snowball_distribution** utility.

  The distribution you specify in the **pin_snowball_distribution** file becomes the default distribution. You can customize this behavior by modifying the policy opcode PCM_OP_SUBSCRIPTION_POL_SNOWBALL_DISCOUNT.
- If the discount is to be distributed unevenly, for example, if each member gets a percentage based on its usage, configure sub-balance contributors. Specify the contributors in the **pin_sub_bal_contributor** file and load the file by running the **load_pin_sub_bal_contributor** utility. See "Defining and Loading Custom Sub-Balances" for information.

Defining How Snowball Discounts Are Distributed

To specify how snowball discounts are distributed, edit the **pin_snowball_distribution** file and then run the **load_pin_snowball_distribution** utility to load the file's contents into the **/config/snowball_distribution** object in the BRM database.

You set two values in the **pin_snowball_distribution** file:

- The name of the discount offer.
- Either zero or the balance element ID for the counter discount.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- If set to zero, the discount is distributed evenly among all members of the discount sharing group.
- If set to a balance element ID, the discount is distributed based on how much each account contributed to the amount in the counter balance.

> **Note**
>
> If you specify a balance element, you must configure sub-balance contributors to enable BRM to track each account's contribution. See "Defining and Loading Custom Sub-Balances" for information.

For example, the following entries in the **pin_snowball_distribution** file specify that **snowball one** is distributed evenly and **snowball two** is distributed based on the amount each account contributed to the counter (balance element ID 1000501) at the end of the billing cycle.

```
snowball one : 0
snowball two : 1000501
```

> **Note**
>
> The **load_pin_snowball_distribution** utility overwrites existing snowball distribution rules. If you are updating snowball distribution rules, you cannot load new snowball distribution rules only. You must load complete sets of snowball distribution rules each time you run the **load_pin_snowball_distribution** utility.

To configure snowball distribution rules:

1. Edit the **pin_snowball_distribution** file in *BRM_home*/**sys/data/pricing/example**. The **pin_snowball_distribution** file includes instructions.
2. Save the **pin_snowball_distribution** file.
3. Use the following command to run the **load_pin_snowball_distribution** utility:

   load_pin_snowball_distribution *pin_snowball_distribution*

   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_snowball_distribution *BRM_home*/**sys/data/pricing/example**

   > **Note**
   >
   > If you copy the **pin_snowball_distribution** file to the directory from which you run the **load_pin_snowball_distribution** utility, you do not have to specify the path or file name. The file must be named **pin_snowball_distribution**.

4. Stop and restart the Connection Manager (CM).

To verify that the network elements were loaded, you can display the **/config/snowball_distribution** object by using the Object Browser, or use the **robj** command with the **testnap** utility. See "Reading an Object and Writing Its Contents to a File" in *BRM Developer's Guide*.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Applying Price Tags to Discount Offers

You can apply special prices to discount offers according to specified rules by using price tags added to balance elements.

> **Note**
>
> You cannot use price tags with temporary balance elements.

To apply price tags to discount offers:

1. Use the PDC UI or the **ImportExportPricing** utility to create price tag setup components. Specify the price tag name and permitted values, as well as balance elements, resource units, and services that the price tag can be associated with.
   For example, you could create a price tag called **FreeMinutesTag** and specify a range of valid values from 10 to 50. You can determine that it will be associated with the Free Minutes balance element, the minute resource unit, and the TelcoGSM and Telephony services.

   BRM stores the pricing tag configuration in the **/config/price_tags** object.

   See "Configuring Price Tags" for more information about creating price tag setup components.
2. Use one of the following to add the price tag, such as **FreeMinutesTag**, to balance elements of discounts:
   - The PDC UI: Click the Price Tag icon in pricing table toolbar and then select a price tag from the Price Tag dialog box. For more information, see "Adding Price Tags for Discounts" in *PDC Online Help*.
   - The **ImportExportPricing** utility: Use the **<priceTag>** element under the discount. See "Applying Price Tags to Discounts".

   When you add a price tag to a balance element, the rules configured in the setup component determine which price tags are available. For balance elements in discount offers and discount selectors, price tags that match the balance element code and service associated with the discount are available.
3. At purchase time, enter the actual value to use in rating for the price tag. The rules configured in the setup component determine which values are valid. For example, if the setup component is configured with the **LIST** rule type, only the values configured in the list in the setup component are valid.
   BRM stores the price tag values in **/offering_override_values** objects.

   You can use BRM opcodes to get the list of valid price tags for a discount offer. See "Getting Information about Price Tags" in *BRM Opcode Guide*.

Tying Discount Offers with Charge Offers

You can tie a discount offer to one or more charge offers within a purchased bundle or package. Use this when a bundle contains multiple charge offers for the same service and event type, but only specific charge offers should receive the discount.

For example, a bundle can include one charge offer for a SIM purchase fee and another charge offer for a SIM activation fee. You can apply a 20% discount only to the SIM activation fee and leave the SIM purchase fee unchanged.

Copyright © 2017, 2026, Oracle and/or its affiliates.

A tie is created by assigning the same tie ID to the purchased discount offer and the purchased charge offer. BRM applies a tied discount offer only to purchased charge offers that have the same tie ID.

To tie discount offers to charge offers:

1. At design time, mark the discount offer as eligible for tying with charge offers. See "Setting Discount Offers as Eligible for Tying ".
2. At purchase time or after purchase, assign the same tie ID to the discount offer and the charge offer.

> **Note**
>
> You can tie multiple discount offers to one purchased charge offer. You can also tie one discount offer to multiple purchased charge offers.

Setting Discount Offers as Eligible for Tying

Before a discount offer can be tied to a charge offer at purchase time, make the discount offer eligible for tying. This enables you to tie discount offers to charge offers at purchase.. Only discount offers marked as tied can be tied with specific charge offers.

You can make a discount offer eligible for tying in either of the following ways:

1. In the PDC UI, when creating or editing a discount offer, select **Tie Discount Offer with Charge Offer** in the Offer Settings section. For more information, see "Specify Discount Offer Settings" in *PDC Online Help*.
2. In the **ImportExportPricing** utility, set the **<tieToChargeOfferEnabled>** element to **true** in the discount offer XML definition.

   For more information, see "Marking Discount Offers as Eligible for Tying with Charge Offers in XML ".

   When this setting is enabled, PDC stores the design-time eligibility in BRM as PIN_FLD_TIED_DISCOUNT. At runtime, the purchased discount offer can then be associated with purchased charge offers by using a tie ID.

Configuring External Applications to Manage Discount Offer Ties

If you want to use an external client application to manage discount offer ties, you must customize it to support the following actions:

- Tie a discount offer to a charge offer. See "Tying Discount Offers at Purchase ".
- Modify the tie ID of a discount offer or charge offer after purchase. See "Changing the Tied Discount for a Charge Offer".
- Handle discount offers that are no longer tied to an active charge offer. See "Handling Untied Discount Offers".
- Retrieve discount offers by tie ID. See "Retrieving Discounts and Purchased Offerings by Tie ID".

For more information, see *REST API Reference for Billing Care*.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Tying Discount Offers at Purchase

To tie a discount offer to a charge offer during purchase, assign the same tie ID to the purchased charge offer and the purchased discount offer. You can do this using the Billing Care UI, BRM opcodes, or the Billing Care REST API.

> **Note**
>
> Use the same tie ID for all purchased charge offers and discount offers that must be tied together. Use a different tie ID for a separate association group.

If you use opcodes, use PCM_OP_SUBSCRIPTION_PURCHASE_PRODUCT and PCM_OP_SUBSCRIPTION_PURCHASE_DISCOUNT. For more information, see "How to Tie a Discount Offer to Charge Offers" in *BRM Opcode Guide*.

If you use the Billing Care REST API, use the endpoints in Table 12-1

Table 12-1 Endpoints for Tying Discount Offers at Purchase

API Endpoint	Request Payload
Create an Account (POST /bcws/webresources/v1.0/accounts)	Use the following request payload fields for tying: • customizedChargeOffers.tieID • customizedDiscountOffers.tieID • customizedDiscountOffers.tieCapability, when overriding the PDC design-time tie eligibility for the purchased discount offer
Add Services to an Account (POST /bcws/webresources/v1.0/accounts/{accountId}/package)	Use the following request payload fields for tying: • customizedChargeOffers.tieID • customizedDiscountOffers.tieID • customizedChargeOffers.tieOfferRefs[] or customizedDiscountOffers.tieOfferRefs[] • customizedDiscountOffers.tieCapability, when overriding the PDC design-time tie eligibility for the purchased discount offer
Add a Bundle to an Account (POST /bcws/webresources/v1.0/accounts/{serviceId}/bundle)	Use the following request payload fields for tying: • customizedChargeOffers.tieID • customizedDiscountOffers.tieID • customizedChargeOffers.tieOfferRefs[] or customizedDiscountOffers.tieOfferRefs[] • customizedDiscountOffers.tieCapability, when overriding the PDC design-time tie eligibility for the purchased discount offer

Changing the Tied Discount for a Charge Offer

After purchase, you can modify the tie ID for a purchased discount offer or purchased charge offer. This lets you associate a different discount offer with a charge offer, associate an existing discount offer with another charge offer, or remove the association. You can do this using the Billing Care UI, BRM opcodes, or the Billing Care REST API.

Copyright © 2017, 2026, Oracle and/or its affiliates.

If you use opcodes, use PCM_OP_SUBSCRIPTION_SET_DISCOUNTINFO and PCM_OP_SUBSCRIPTION_SET_PRODINFO. For more information, see "How the Discount Offer and Charge Offer Tie ID Is Modified" in *BRM Opcode Guide*.

If you are using the Billing Care REST API, use the endpoints in Table 12-2.

Table 12-2 Endpoints for Modifying Discount Offer and Charge Offer Tie ID

API Endpoint	Request Payload
Modify a Purchased Product Offer (PUT /bcws/webresources/v1.0/subscriptions/offerdetails/{purchasedProductId})	Use the following request payload fields to modify tying: • customizedChargeOffers.tieID • customizedChargeOffers.tieOfferRefs[], when the request ties the purchased charge offer by offer reference
Modify a Purchased Discount Offer (PUT /bcws/webresources/v1.0/subscriptions/offerdetails/{purchasedDiscountId})	Use the following request payload fields to modify tying: • customizedDiscountOffers.tieID • customizedDiscountOffers.tieOfferRefs[], when the request ties the purchased discount offer by offer reference • customizedDiscountOffers.tieCapability, when overriding the PDC design-time tie eligibility for the purchased discount offer

To remove an association, clear the tie ID for the purchased offer. The Billing Care UI displays a purchased offer that does not have a tie ID as **Not Set**.

Handling Untied Discount Offers

An untied discount offer is a purchased discount offer that has the capability enabled but is no longer tied to an active purchased charge offer.

When a discount offer becomes untied, you can do any of the following:

- Reuse the same tie ID to tie the discount offer to a different charge offer in the bundle.
- Set a different tie ID to tie the discount offer to another charge offer.
- Clear the tie ID and leave the discount offer not tied.
- Configure BRM to automatically cancel discount offers that are not tied to any charge offer.

For more information, see "Managing Untied Discount Offers" in *BRM Opcode Guide*.

Retrieving Discounts and Purchased Offerings by Tie ID

To retrieve information about discounts and purchased offerings tied with a specific tie ID, use BRM opcodes or the Billing Care REST API.

If you use opcodes, you can retrieve eligible discounts for a product or list all discounts and charge offers already tied with an account or service. For more information, see "Retrieving Purchased Offerings for an Account or Service" in *BRM Opcode Guide*.

If you use the Billing Care REST API, refer to the endpoints in Table 12-3

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 12-3 Endpoints for Retrieving Discounts and Purchased Offerings by Tie ID

API Endpoints	Response Payload
Retrieve a Purchased Charge Offer or Purchased Discount Offer (GET /bcws/webresources/v1.0/subscriptions/offerdetails/{purchasedProductId})	Use the following response payload fields to retrieve tying information: • customizedChargeOffers.tieID • customizedDiscountOffers.tieID • customizedDiscountOffers.tieCapability
Retrieve Customized Offers for an Account (GET /bcws/webresources/v1.0/subscriptions/account/{purchasedProductID}/customizedOffers)	Use the following response payload fields to retrieve tying information: • customizedChargeOffers.tieID • customizedDiscountOffers.tieID • customizedDiscountOffers.tieCapability
Retrieve Customized Offers for a Service (GET /bcws/webresources/v1.0/subscriptions/service/{serviceId}/customizedOffers)	Use the following response payload fields to retrieve tying information: • customizedChargeOffers.tieID • customizedDiscountOffers.tieID • customizedDiscountOffers.tieCapability

Copyright © 2017, 2026, Oracle and/or its affiliates.

13

Creating Subscription Terms

Learn how to create subscription terms and add them to your packages and bundles in Oracle Communications Billing and Revenue Management (BRM). Subscription terms define the commitment periods and options for canceling and renewing the services you offer.

Topics in this document:

- About Subscription Terms
- About Penalty Fees
- About Automatic Renewals
- About Bundle Terms and Package Terms
- About Obsoleting Subscription Terms
- Creating Subscription Terms

About Subscription Terms

Subscription terms specify a contract's commitment period, such as 1 month, 1 year, or 2 years.

Terms 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 or the contract balance
- Have a grace period, during which the penalty is waived
- Have their contracts renewed automatically when the commitment period ends
- Have a minimum contract period before cancellation is permitted

For example, a term might specify a 1-year commitment period, a minimum contract length of 4 months, allow early cancellation with a $250 early termination fee, and renew automatically with the same terms.

After you create subscription terms, you can associate them with your packages and bundles.

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 13-1 describes the types of charges you can apply.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 13-1 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.
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. • 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 automatically renew at the end of the commitment period with the same or different terms. For example, you can create a subscription term that automatically renews with the same one-year commitment or with a different monthly commitment.

Time-based discounts, such as a first-year discount, aren't renewed when a contract is 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 manually renew their contracts before they expire, 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

Copyright © 2017, 2026, Oracle and/or its affiliates.

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.

About Obsoleting Subscription Terms

> **Note**
>
> When a package 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 package and the bundles associated with the package all use the same subscription term. Before you can obsolete the term, you must change the package and its bundles 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 auto-renew setting to **No**.

Creating Subscription Terms

You can create subscription terms using:

- The PDC UI. See "Subscription Terms" in *PDC Online Help*.
- The **ImportExportPricing** utility with an XML file. For information about the XML tags used to configure subscription terms, see the *PDC_home***/apps/Samples/Examples/Sample_SubscriptionTerms.xml** file and "Subscription Terms".

When creating terms, you specify the following:

- The commitment period.
- Whether to allow early cancellation of the contract and whether to apply a penalty fee:
  - **Allowed, No Penalty:** Contracts can be canceled early, and customers will not be charged a penalty.
  - **Allowed, with Penalty:** Contracts can be canceled early, but customers will be charged a penalty fee for doing so.
  - **Not Allowed:** Contracts cannot be canceled early.
- If applying a penalty for early cancellation, whether to apply an early termination fee, the contract balance, or both.
  - If applying an early termination fee, whether it's a flat fee or a fee that reduces over time.
  - If applying the contract balance, whether to apply the remaining balance up to the full commitment period or up to a minimum contract length.

  You can also specify:

Copyright © 2017, 2026, Oracle and/or its affiliates.

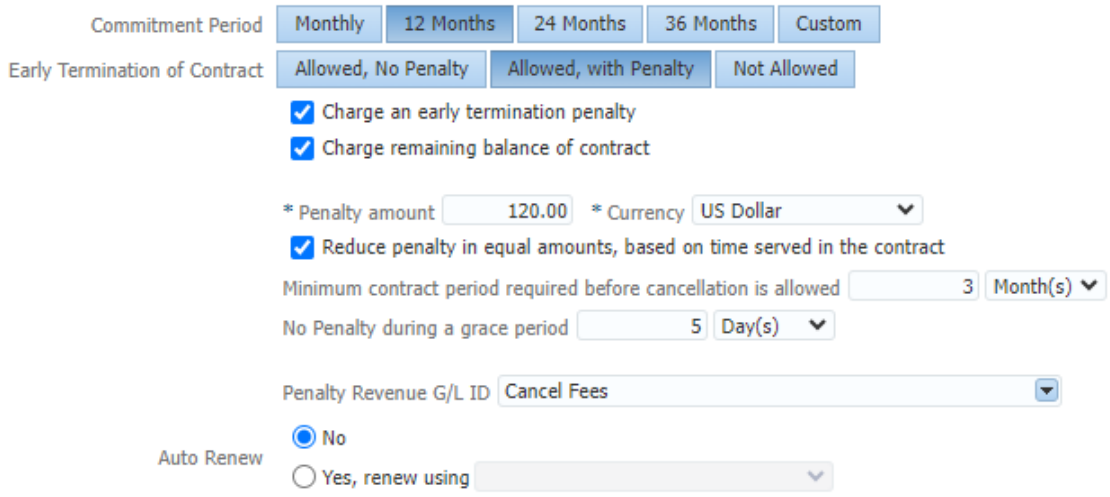
  - Whether to add a grace period during which no penalty is applied.
  - The revenue general ledger ID to associate with the early termination fee.

    If you do not select a general ledger ID, the early termination fee is distributed across the deliverables in the contract. This applies only if you use deliverable-based revenue recognition.
- Whether to automatically renew the contract, and which terms to renew with.
- The minimum contract period before cancellation is permitted.

After you have created terms, they can be associated with your packages and bundles.

Figure 12-1 shows how to use the PDC UI to create a sample subscription term that has a 12-month commitment period, allows early cancellation with a $120 early termination fee that reduces by $10 each month, has a 5-day grace period, and does not renew automatically. The minimum contract period before cancellation is permitted is 3 months.

Figure 13-1 Early Termination Fee Settings in PDC



Copyright © 2017, 2026, Oracle and/or its affiliates.

14

Creating Bundles

Learn how to use bundles to group charge offers and discount offers for a single service you want to sell together in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

- About Creating Bundles
- Configuring Bundle Settings
- Associating Terms with Bundles
- Customizing Offers In Bundles
- Creating Dependencies for Bundles
- Transitioning between Bundles

About Creating Bundles

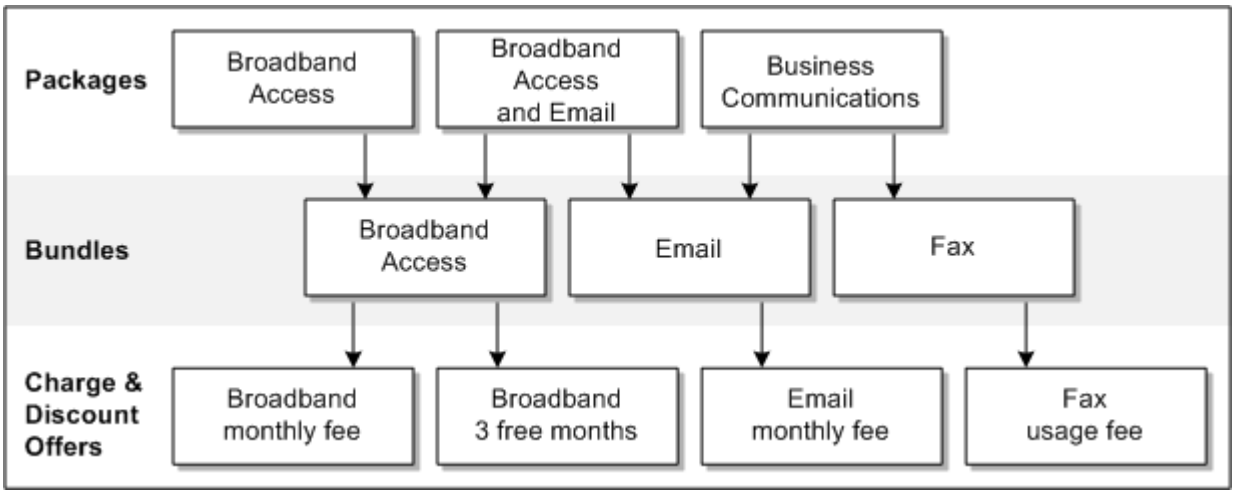
A bundle is a set of charge offers, discount offers, or both. You typically use bundles to group offers that you want to sell together.

Each bundle is associated with a single service. Only offers that apply to that service can be included in the bundle. For example:

- A package that provides broadband access includes only broadband access bundles (such as cable and premium cable).
- A package that provides broadband access and VOIP includes at least two bundles, one for broadband access and one for VOIP.
- A package that provides VOIP and cable TV includes at least two bundles, one for VOIP and one for cable TV.

Figure 14-1 shows bundles that are associated with a single service.

Figure 14-1 Bundles Are Always Associated with a Single Service



Copyright © 2017, 2026, Oracle and/or its affiliates.

One bundle can contain any number of charge offers and discount offers, and two different bundles can contain the same offer. Grouping offers in different ways in different bundles adds flexibility to your pricing structure without requiring you to create additional charge offers or discount offers.

Creating a bundle involves these high-level tasks:

1. Configuring the bundle's validity dates, the service or account it is associated with, and other settings.

   See "Configuring Bundle Settings".

2. Optionally, adding subscription terms. Subscription terms govern the agreement between you and the customer who purchases the bundle. When customers purchase a bundle with a subscription term, it becomes a contract.

   See "Associating Terms with Bundles".

3. Adding one or more charge offers and discount offers.

4. Optionally, customizing the charge offers and discount offers.

   See "Customizing Offers In Bundles".

5. Optionally, defining any dependencies between this bundle and other bundles.

   See "Creating Dependencies for Bundles".

6. Optionally, specifying whether customers can transition from this bundle to another specified bundle.

   See "Transitioning between Bundles".

7. Optionally, configuring PDC to save extra information about the bundle that may be useful to external applications.

   See "Configuring Product Specification Attributes for Pricing Components".

Configuring Bundle Settings

You can configure bundles in the following ways:

- **Associate the bundle with a service or an account.** When creating a bundle, you can associate it with a specific service. The offers in the bundle only apply to that service.

  Alternatively, you can associate the bundle with **Account**. You might do this, for example, to create a bundle for late charges or coupons. The offers in the bundle can then be used to rate any event associated with **Account** in the service-event map. Each account can own only one bundle that applies to **Account**.

- **Bill customers when they purchase the bundle.** Usually, you bill a customer at the end of the customer's billing cycle. *On-purchase billing,* however, enables you to bill a customer immediately for a purchase, even if the customer's billing cycle has not ended. When you create a bundle, you can flag it for on-purchase billing. When a customer purchases the bundle, a bill is generated immediately for the purchase fees associated with the bundle.

> **Note**
>
> On-purchase billing only works with purchase fees, not recurring, usage, or cancellation fees.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **Align balance impacts on the first usage**. You can configure a bundle to synchronize the start date of all balance impacts whose validity period starts on first usage. This ensures that all balance impacts in the bundle's charge offers are set to the same validity period when one of them is activated for the first time.

  For information about how it affects discount offers, see "About Activating First-Usage Discount Offers".

  For information about how it affects charge offers, see "About Granting First-Usage Balance Impacts In First-Usage Charges".

- **Permit CSRs to customize bundles.** If your CSRs (customer service representatives) can discount or change the effective period of charges in bundles at purchase time, you can specify whether to prohibit, allow, or require such modifications in a particular bundle. For example, if customer input is required to set the date on which an offer's purchase fee is applied, you can specify that a bundle must be modified.
- **Specify when a bundle can be purchased.** By default, the period during which a bundle can be purchased starts immediately and never ends. You can change the default start date, end date, or both.

  To be added to a bundle, an offer must have a purchase period equal to or greater than the bundle purchase period. If the purchase period of an offer in a bundle exceeds the bundle purchase period, the bundle purchase period overrides the offer purchase period.

About Activating First-Usage Discount Offers

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

In some cases, a discount's balance impact is negated by a second discount. If this occurs, the validity period of the first discount is still set.

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

- Discount offer 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 offer 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 offer A is applied first, reducing the balance by $.50, and discount offer A's validity period is set. Then discount offer B backs out all charges. In this case, the validity period of discount offer A remains set.

About Granting First-Usage Balance Impacts In First-Usage Charges

A charge offer valid from first usage should not grant a credit balance, such as included minutes, that is also valid from first usage. If it does, the first usage event cannot consume the granted balance.

Copyright © 2017, 2026, Oracle and/or its affiliates.

For example, consider a charge offer that is valid on first usage and has these charges:

- A recurring event that grants minutes. The balance impact is set to the first usage.
- A charge that consumes minutes until there are none and then consumes money.

When the first usage event for this charge offer occurs, the charges are processed in this order:

1. The first usage event is charged, and it consumes a monetary balance because no minutes have been granted.
2. The recurring event, with a balance impact set to the first usage, can now be charged. It grants minutes.
3. Subsequent usage events now have minutes to consume.

Because the included minutes are valid from the first usage, they are not granted until after the first usage event is rated. Hence, the first usage event cannot consume any of the minutes included in the charge offer. Therefore, the customer is charged for a call that should have consumed minutes.

However, if a discount offer that is valid from first usage grants a credit balance that is valid at first usage, the granted balance is available for consumption because discounting occurs after the balances are granted.

Associating Terms with Bundles

You can associate your bundles with subscription terms, which govern the agreement between you and the customer who purchases the bundle. Subscription terms specify the bundle's commitment period, whether customers are allowed to cancel the bundle early, and whether customers are charged an early termination fee for doing so.

If you add a subscription term to a bundle, it applies only if the bundle is *optional* in a package.

If a package specifies that the bundle is *required*, the bundle's terms will be overridden by the package's terms. See "About Bundle Terms and Package Terms".

Customizing Offers In Bundles

You can customize the charge offers and bundle offers in a bundle in the following ways:

- **Customize the offer quantity**. You can customize bundles to provide more than one of the same charge offer or discount offer. For example, if a bundle for a cable service includes a charge offer that provides one set-top box and you want to include three set-top boxes with the cable service, enter 3 for the charge offer in that bundle.
- **Override the offer's validity period**. You can override the validity start and end dates in a charge offer or discount offer, such as from January through March to a new validity period of January through July. See "About the Validity Periods of Offers in Bundles".
- **Set the status of offers at purchase**. You can specify whether an offer is *active* or *inactive* at the time of purchase. For example, an offer might be inactive at purchase so that the purchase or first month's fee is only applied after you get confirmation that the hardware was received and successfully configured. By default, offers are active.

  For offers with inactive status, you can specify a reason for that status.
- **Configure how to set a charge offer's cycle alignment when its bundle is reactivated**. Suppose a customer suspends and later reactivates their subscription. In that case, you

Copyright © 2017, 2026, Oracle and/or its affiliates.

can specify whether the charge offer's cycle alignment is based on the original billing, purchase, or reactivation date.

By default, the cycle is aligned with the original billing or purchase dates, depending on how the charge's cycle alignment is configured. You can instead select the **Renewal Mode** option for the charge offer to use the reactivation date instead of the original date. When you select this option, the following scenarios are possible:

- If the charge's cycle is set to align with the purchase date, the reactivated charge offer's cycle aligns with the reactivation date.
- If the charge's cycle is set to align with the purchase date, but the charge offer specifies that the cycle alignment should be a specific day of the month, the reactivated charge offer's cycle aligns with whichever is later, the reactivation date or the day of the month specified in the charge offer.
- If the charge's cycle is set to align with the billing date, the reactivated charge offer's cycle will align with the billing date (regardless of whether you select the **Renewal Mode** option).

- **Configure how to prorate an offer's cycle fees**. You can specify to prorate an offer's fees using a 30-day month or the actual number of days in the month. See "Setting Proration for Offers in a Bundle".
- **Specify whether a charge offer can be purchased as an add-on product**. You can specify that a charge offer is optional, meaning it can be purchased with the bundle or later. See "About Add-On Charge Offers in Bundles".
- **Configure what happens when a customer purchases the same offer more than once**. You can specify whether the additional offer is purchased as a new subscription or as a replacement. See "About Purchasing the Same Offer Multiple Times".
- **Customize charges and discounts for a customer**. See "About Customizing Charges and Discounts in Bundles".

About the Validity Periods of Offers in Bundles

You can configure the start time and end time for each charge offer and discount offer in a bundle. The time between the start time and the end time is called the *validity period*. A customer cannot use a service provided by an offer unless the offer is valid.

> **Note**
>
> - You specify when a charge offer or a discount offer can be **purchased** in the offer itself. You cannot apply a validity period when you create a charge offer or a discount offer; you can only apply it in the bundle configuration.
> - You can apply validity periods to individual charges and discounts configured within offers. However, the validity periods specified in a bundle take precedence over the validity periods specified in the charges and discounts. For example, you might want to allow a customer to purchase a charge offer but not allow the customer to use the service until it is activated on the network.

In bundles, you set the validity period of offers by specifying the start and end times of the offer and of the offer's cycle and usage charge periods.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **Offer start and end times:** Specify when a customer can use the service or benefit from the discount. The offer start time is when the purchase fee is charged. It is also the earliest time that the offer's fees can begin to accumulate in an account balance.
- **Recurring and usage charge periods:** Specify when recurring and usage events can be charged or discounted. These validity periods must not begin before the offer start time.

You can set *start* times as follows:

- **Immediately:** (Default) The offer or charge is valid and can be activated immediately. The purchase fee is charged as soon as the offer is added to the account.
- **Relative to Purchase:** The purchase fee is charged when the offer is added to the account, but the customer cannot use the service or benefit from the discount until the relative period ends.

  When a charge offer's charge period has a relative start time, the events are not rated, and the fees are not charged 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 of time.

  When a discount offer's charge period has a relative start time, the discount is not applied to cycle or usage fees until the relative period ends.
- **First Usage of Any Offer in a Bundle:** When a customer first uses any service in a bundle, all offers in the bundle are activated and all purchase fees are charged. The charge periods also begin at that time. The validity period start time is set to the start time of the event that first uses the service or triggers the discount offer.

  Setting charge offers and discount offers 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 discount offers until they can be applied to customers' usage. This is useful when you offer limited-time services or discount offers that expire relative to when they are activated.

> **Note**
>
> Charge offers that start on first usage must include usage fees.

You can set the *end* times as follows:

- **Never:** (Default) After it is activated, the offer is effective indefinitely, and its recurring and usage fees can be charged or discounted indefinitely.
- **Relative to Start:** After the relative period ends, the offer is not effective, and the recurring or usage fees are not charged or discounted.

If the offer end time specified in the bundle is earlier than the recurring or usage charge end time, the offer end time overrides the charge end time.

Setting Proration for Offers in a Bundle

When creating bundles, you can specify how cycle charges and discounts are prorated at the charge offer or discount offer level. You can specify to calculate prorated cycle charges and discounts using:

- **A 30-day month**: Prorated cycle charges and discounts are calculated based on a 30-day month, regardless of the number of days in the billing cycle. For example, if a bundle was owned for six days in a cycle, the prorated charge would be the cycle charge multiplied by 0.20 (6 ÷ 30).

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **The actual number of days in a billing cycle**: Prorated cycle charges and discounts are calculated based on the actual number of days in a particular cycle, such as 28 days in February, 31 days in March, and 30 days in April. For example, if a bundle was owned for six days in March, the prorated charge would be the cycle charge multiplied by 0.19 (6 ÷ 31).
- **The system-wide proration setting**: Prorated cycle charges and discounts 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*.

You can configure proration at the offer level in your bundles by using the following:

- The PDC UI. When customizing an offer in your bundle, use the **Proration** list. For more information, see "Customizing an Offer in a Bundle" in *PDC Online Help*.
- The **ImportExportPricing** utility. In your XML file, set the **<prorationBasedOn>** element under **<bundledProductOfferingItem>**. For more information, see "Configuring Proration for Offers in Bundles".

About Add-On Charge Offers in Bundles

You can include add-on charge offers in your bundles. All charge offers in bundles are base charge offers by default, which means they can be purchased without any prerequisites. Add-on charge offers require customers to own a valid base charge offer.

When you create an add-on charge offer, you specify how to determine its validity start date. The start date of the add-on charge offer's validity is the end date of the charge offer that you specified. For example, assume charge offer A has a validity period from June 1 through June 15. If you specify to align add-on charge offer B's validity period with charge offer A, charge offer B's validity start date would be June 15.

You can specify that an add-on charge offer's validity dates align with:

- The base charge offer that you specify
- The active base charge offer that expires first
- The active base charge offer that expires last
- The active charge offer that expires first
- The active charge offer that expires last

You configure an add-on charge offer in a bundle by using the following:

- The PDC UI. When customizing a charge offer in a bundle, set the **This is an add-on charge offer** option. For more information, see "Customizing an Offer in a Bundle" in *PDC Online Help*.
- The **ImportExportPricing** utility. In your XML file, set the **<addOnOfferingValidityMode>** element under **<bundledProductOfferingItem>**. For more information, see "Configuring Add-On Charge Offers In Bundles".

About Purchasing the Same Offer Multiple Times

You can specify what happens if customers purchase the same charge offer or discount offer more than once.

For both charge offers and discount offers, you can specify whether the additional offer is purchased as a new subscription, a replacement of the existing subscription, or an extension of the existing subscription:

Copyright © 2017, 2026, Oracle and/or its affiliates.

> **⚠ Caution**
>
> The following values can be applied to one-time charges only: **OVERWRITE**, **LONGEST_DATE**, **LONGEST_DATE_NEW_SUB_BAL**, **EXTEND**, and **EXTEND_NEW_SUB_BAL**. These values cannot be applied to cycle or usage charges.

- These options create a new subscription or replace an existing one:
  - **Create a new subscription and keep the old one**: The additional charge offer or discount offer is purchased as a new, unrelated subscription. The balance impacts and validity periods of the old and new subscriptions are entirely independent. This option is the default.
  - **Create a new subscription and cancel the old one**: The additional charge offer or discount offer replaces the existing one. The existing subscription is canceled, any configured proration is applied, and the new subscription is created, with all balance impacts and validity periods set as though for a new purchase.

    For example, on June 1, a customer purchases an add-on bundle that grants 3 GB of data and is valid for seven days. On June 3, after using 1 GB, they repurchase the bundle. The old charge offer is canceled, any remaining balance is prorated, and the new 3 GB balance is created, with a validity end date of June 10.
- These options extend the old subscription if the additional charge offer is purchased within the specified grace period. The grace period can be any number of seconds, minutes, hours, or days. The default grace period is 0 days.
  - **Extend the old subscription, setting validity to whichever is later, the new or existing validity**: The new balance impacts are added to the existing balance group. The balance's validity period is 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 an add-on bundle that grants 3 GB of data, is valid for seven days, and has a grace period of four days. On June 3, after using 1 GB, they purchase the bundle again. The new 3 GB balance is added to the remaining balance for a total of 5 GB. BRM compares the validity periods of the old and new charge offers and sets the validity end date for the total balance to the later of the two end dates, June 10
  - **Extend the old subscription with a separate balance, setting validity to whichever is later, the new or existing validity**: The new balance impacts are added to a new balance group. The new balance's validity period starts on the purchase date and ends either on the old or new validity end date, whichever is later.

    For example, on June 1, a customer purchases an add-on bundle that grants 3 GB of data, is valid for seven days, and has a grace period of four days. On June 3, after using 1 GB, they purchase the bundle again. The new 3 GB balance is added to a new balance group, separate from the remaining 2 GB in the old balance group.

    BRM calculates the validity end date by comparing the validity periods of the old and new charge offers and setting the validity end date for the new balance to the later of the two end dates, June 10. The old balance retains the original validity dates.
  - **Extend the old subscription, adding the new validity to the old validity**: The new balance impacts are added to the existing balance group. The balance's validity period is set by adding the new validity period to whatever remains of the old validity period.

Copyright © 2017, 2026, Oracle and/or its affiliates.

For example, on June 1, a customer purchases an add-on bundle that grants 3 GB of data, is valid for seven days, and has a grace period of four days. On June 3, after using 1 GB, they purchase the bundle again. The new 3 GB balance is added to the remaining balance for a total of 5 GB. BRM adds the validity period of the new charge offer to the old charge offer and sets the validity end date for the total balance to June 15.

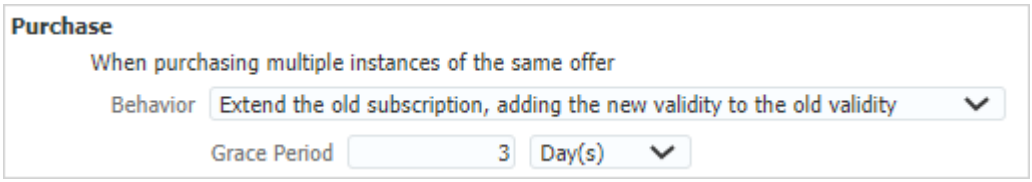
- **Extend the old subscription with a separate balance, adding the new validity to the old validity**: The new balance impacts are added to a new balance group. The new balance's validity period starts on the end date for the old balance's validity, and the end is set by adding any remaining validity from the old balance to the total validity of the new balance.

  For example, on June 1, a customer purchases an add-on bundle that grants 3 GB of data, is valid for seven days, and has a grace period of four days. On June 3, after using 1 GB, they purchase the bundle again. The new 3 GB balance is added to a new balance group, separate from the remaining 2 GB in the old balance group.

  The new balance is valid on June 8. BRM calculates the new balance's validity end date by adding the validity period of the new charge offer to the old charge offer and setting the validity end date for the new balance to June 15. The old balance retains the original validity dates.

You configure how to handle multiple purchases of the same offer by using one of the following:

- The PDC UI. To do so, use the **Behavior** list and **Grace Period** fields when customizing the offers in your bundles. For more information, see "Customizing an Offer in a Bundle" in *PDC Online Help*.



- The **ImportExportPricing** utility. To do so, set the **<purchaseMode>**, **<gracePeriodOffset>**, and **<gracePeriodUnit>** elements in your import XML file. For more information, see "Purchasing the Same Offer in a Bundle Multiple Times".

About Customizing Charges and Discounts in Bundles

In addition to adding discount offers to a bundle, you can reduce one-time and recurring charges in charge offers by specifying a flat percentage discount for each charge category in the bundle itself. For example, you can discount all recurring charges and one-time charges by 10%. You cannot use a bundle to apply discounts to usage charges.

When you discount a charge category in a bundle, the discount applies to all charge types that apply to that charge category. For example, you might have a charge offer that includes two recurring charges: Monthly Fee and Grant Minutes. If you specify a percentage discount for recurring charges in the bundle, the discount applies to both recurring charges. To discount only the Monthly Fee charge, you must make the balance impact for Monthly Fee discountable and the balance impact for Grant Minutes nondiscountable.

Unlike discount offers, discounts configured in bundles cannot be tracked in the general ledger because they are not associated with a general ledger (G/L) ID. In addition, unlike discount

Copyright © 2017, 2026, Oracle and/or its affiliates.

offers, bundle-configured discounts are difficult to display on a customer's bill because they are not separate items.

> **Note**
>
> Because of the limitations associated with discounts configured in bundles, Oracle recommends that you use discount offers instead. See "Creating Discount Offers" for more information.

Creating Dependencies for Bundles

You can set up the following dependent relationships between bundles:

- **Prerequisite:** Specifies that an account must own a particular bundle to be able to purchase another bundle. A prerequisite can include bundles for different services. For example, to own a GPRS bundle, an account might be required to own a GSM bundle.
- **Mutually Exclusive:** Sets up a mutually exclusive relationship between two bundles so that if an account owns one of the bundles, it cannot own the other. For example, if you set up a mutually exclusive relationship between a Corporate Voice bundle and a Residential Voice bundle, customers who purchase one cannot purchase the other.

Transitioning between Bundles

You can configure rules for transitioning from one bundle to another. PDC supports the following types of transitions:

- **Upgrade** to a bundle that is typically more expensive and has more features.
- **Downgrade** to a bundle that is typically less expensive and has fewer features.

Transition rules enable you to limit the bundles that customers can transition to and remain fully provisioned. While transitioning from one bundle to another, your customers retain their devices, such as phone numbers, and their services.

> **Note**
>
> Transitioning between bundles affects only the charge offers and discount offers related to the service to which the bundle applies; the service itself remains the same. To define a transition that adds a new service, you create a transition for a package. See "Transitioning Between Packages" for more information.

For information about customizing bundle transitions, see "Customizing Bundle Transitions" in *BRM Opcode Guide*.

Prorating Charges During Bundle Transitions

You can also configure how PDC handles charges when customers transition bundles in the middle of their billing cycle. Charges can be applied from the original bundle, applied from the new bundle, or prorated for both bundles.

Copyright © 2017, 2026, Oracle and/or its affiliates.

For example, assume a customer's billing day of the month (DOM) is the 15th, and on June 30 he transitions from Bundle A to Bundle B. If you configure PDC to:

- Prorate the charges, the customer's July 15 bill would include charges for Bundle A prorated from June 15 through June 30, and charges for Bundle B prorated from July 1 through July 14. This is the default.
- Apply the original bundle's charges, the customer's July 15 bill would include charges for Bundle A only.
- Apply the new bundle's charges, the customer's July 15 bill would include charges for Bundle B only.

You configure how to prorate charges during bundle transitions by using the following:

- The PDC UI. For information, see "Defining Transition Rules for Bundles" in *PDC Online Help*.
- The **ImportExportPricing** utility. In your input XML file, set the **<prorationType>** element under the **<bundleTransition>** element to one of the following:
  - **PRORATE_CHARGE**: Specifies to prorate charges for both bundles. This is the default.
  - **ORIGINAL_CHARGE**: Specifies to apply the full charges from the original bundle.
  - **TRANSFER_CHARGE**: Specifies to apply the full charges from the new bundle.

  For information, see "Transitioning Bundles".

Copyright © 2017, 2026, Oracle and/or its affiliates.

15

Creating Packages and Package Lists

Learn how to create packages and package lists to offer your services to customers in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

- Creating Packages
- Associating Terms with Packages
- Grouping Services
- Transitioning Between Packages
- Creating Package Lists

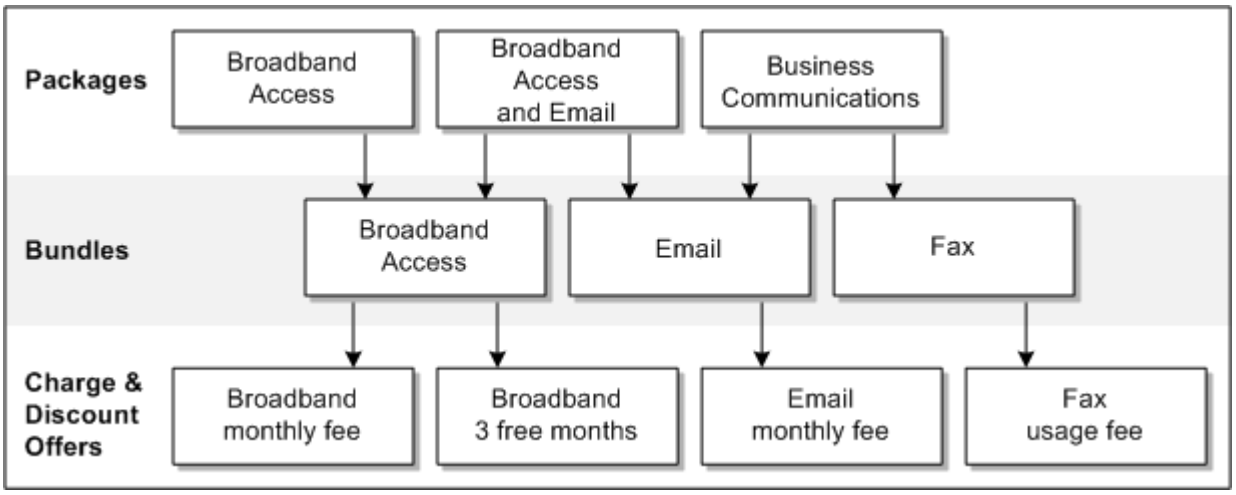
Creating Packages

A package is a collection of bundles. You use packages to offer your services to customers. For example, if your company sells broadband access, email, and IP fax service, you might offer the following packages:

- A package that sells only broadband access
- A package that sells broadband access and email
- A package that sells email and IP fax service

Figure 15-1 shows how you can include bundles and the services to which they apply in a variety of packages.

Figure 15-1 A Bundle Can Be Included in Multiple Packages



A package can contain any number of bundles, and two different packages can share the same bundle. By grouping bundles into packages, you simplify the choices presented to your customers.

Copyright © 2017, 2026, Oracle and/or its affiliates.

You have the following options when creating packages:

- **Bill customers when the package is purchased.** Usually, you bill a customer at the end of the customer's billing cycle. On-purchase billing, however, enables you to bill a customer immediately for a purchase, even if the customer's billing cycle has not ended. When you create a package, you can flag it for on-purchase billing. When a customer purchases a package that is flagged for on-purchase billing, a bill is generated immediately for the purchase fees associated with the package.

  > **Note**
  >
  > On-purchase billing works with purchase fees only. It does not work with recurring, usage, or cancellation fees.

- **Activate all offers in the package when the first service is used.** You can specify to activate all offers in a package when any of the offers are activated on first usage. For example, assume a package includes bundle 1 with offers A and B, and bundle 2 with offer C. If the first service a customer uses is in charge offer B on 15 May, all offers (A, B, and C) are activated on 15 May.
- **Configure required and optional bundles in a package.** *Required bundles* are automatically purchased when a package is purchased. By default, all bundles are required. *Optional bundles* can be purchased at a later time.
- **Restrict discounts in a package.** When creating a package, you can prohibit specified discount offers from being purchased or used while the package is owned.

  If an exclusion relationship exists between a discount offer and a package, a customer can own the discount offer or the package, but not both. Further, the customer cannot own any discount offers associated with the package if the customer owns the excluded discount offer.

  See "Configuring Mutually Exclusive Discount Offers" for information about configuring discount exclusion.
- **Set values for any product specification attributes defined in XML templates.** See "Configuring Product Specification Attributes for Pricing Components".

See the following topics for more package configuration features:

- Associating Terms with Packages
- Grouping Services
- Transitioning Between Packages

Associating Terms with Packages

You can associate your packages with subscription terms, which govern the agreement between you and the customer who purchases the package. When a customer purchases a package with subscription terms, it becomes a contract.

Subscription terms specify the commitment period, whether customers are allowed to cancel their contract early, and whether customers are charged an early termination fee for doing so.

A package's terms apply to all required bundles in the package, but any optional bundles are governed by their own, independent terms. See "About Bundle Terms and Package Terms".

Copyright © 2017, 2026, Oracle and/or its affiliates.

Grouping Services

In packages, you can combine services into service groups. A service group consists of one subscription service, and one or more member services. For example, if the subscription service is GSM, the member services could be Voice and SMS.

Service groups provide the following benefits:

- Member services can benefit from discounts owned by the subscription service.
- Member services can be associated with the devices owned by the subscription service.
- Customers can purchase services as a group.

The subscription service can be a service that subscribers use, such as telephony or broadband access, or it can be a representational service with no associated usage fees. Together, the subscription service and member services form a *service group*.

For example, in Figure 15-1, the GSM service is a subscription service, and SMS and Voice are its member services.

Grouped services usually apply to a particular device, such as a cable box. An account can contain multiple service groups.

To subscribe to a service group, customers subscribe to its subscription service.

Transitioning Between Packages

You can configure rules for transitioning from one package to another. Each transition rule applies to a particular service. The package being transitioned to, and the package being transitioned from must both contain that service, though each package can contain additional services that the other package does not contain.

PDC supports the following types of transitions:

- **Upgrade** to a package that is typically more expensive and has more features. For example, a customer might transition from a package that provides broadband and cable TV services to a package that provides broadband, cable TV, and VOIP services. That transition adds a service and a bundle and possibly changes the bundles for the existing services.
- **Downgrade** to a package that is typically less expensive and has fewer features.

Transitions enable you to limit the packages that customers can switch to while remaining fully provisioned. When transitioning to a designated package, your customers retain their devices, such as phone numbers, and their services.

Prorating Charges During Package Transitions

You can also configure how PDC handles charges when customers transition packages in the middle of their billing cycle. Charges can be applied from the original package, applied from the new package, or prorated for both packages. For example, assume a customer's billing day of month (DOM) is the 15th and on June 30 he transitions from Package A to Package B. If you configure PDC to:

- Prorate the charges, the customer's July 15 bill would include charges for Package A prorated from June 15 through June 30, and charges for Package B prorated from July 1 through July 14. This is the default.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- Apply charges from the original package, the customer's July 15 bill would contain charges for Package A only.
- Apply charges from the new package, the customer's July 15 bill would contain charges for Package B only.

You configure how to prorate charges during package transitions by using the following:

- The PDC UI. See "Defining Transition Rules for Bundles" in *PDC Online Help*.
- The **ImportExportPricing** utility. In your input XML file, set the **<prorationType>** element under the **<packageTransition>** element to one of the following:
  - **PRORATE_CHARGE**: Specifies to prorate charges for both packages. This is the default.
  - **ORIGINAL_CHARGE**: Specifies to apply the full charges from the original package.
  - **TRANSFER_CHARGE**: Specifies to apply the full charges from the new package.

For information, see "Transitioning Bundles".

Defining Generation Changes

A generation change enables you to transition customers between 2G (second generation) and 3G (third generation) wireless packages and services. Packages are called 2G or 3G depending on whether the wireless service they provide runs on a 2G or 3G wireless network.

When creating a 2G or 3G package, you can set up generation change rules to specify which packages can replace the package when it is transitioned to a different generation.

For generation changes, the packages do not have to provide the same service.

> **Note**
>
> The same package cannot be used in both a transition rule and a generation change.

Creating Package Lists

A package list is a group of packages that is usually offered to a single type of customer. For example, you might create the following package lists:

- A package list that includes packages for customers above a certain age.
- A package list that includes packages for customers in a particular location, such as Canada.
- A package list that includes promotional discounts offered for a limited time.

You can create any number of package lists, and each package list can contain any number of packages. Different package lists can contain the same package.

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

To make a package available for customers to purchase, you must include the package in a package list.

You can assign the following statuses to a package list:

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **Active:** Use this status to indicate that customers can purchase packages in the list as soon as the list is added to your system.
- **Inactive:** Use this status to indicate that the list is not visible to customers and customers cannot purchase packages from the list.

Specifying the Package List Segment and Type

The package list segment identifies the customer segment that the package list is offered to, and the package list type identifies the type of packages that are added to the package list.

The package list segment and type determine the package list name. By default, the package list segment must be **CSR**. The package list type can be either **New** or **Add-on**. For example:

- The **CSR-New** package list contains packages that create accounts and add the services and bundles in the packages to the new accounts.
- The **CSR-Add-on** package list contains packages that add services and bundles to existing accounts.

The package list segment and type are case sensitive and together uniquely identify package lists. For example, **CSR-New** and **CSR-new** are two different package lists.

Copyright © 2017, 2026, Oracle and/or its affiliates.

16

Sharing Charges and Discounts

Learn how to set up charge sharing for Oracle Communications Billing and Revenue Management (BRM) in Pricing Design Center (PDC).

Topics in this document:

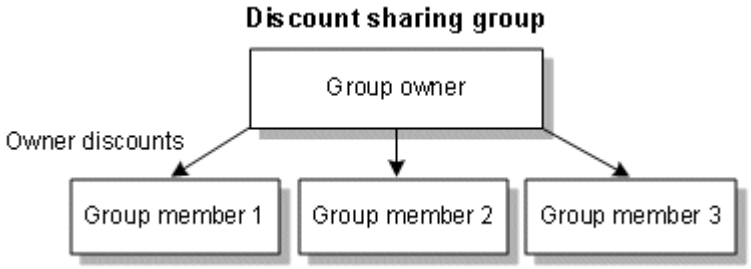
- About Sharing Discounts and Charges
- About First-Usage Start Time for Shared Balances
- About Discount Sharing
- About Chargeshare Offers

About Sharing Discounts and Charges

BRM enables accounts to share discounts and charges by joining groups that consist of an owner account and one or more member accounts. A sharing group can be one of the following types:

- **Discount sharing group:** The owner shares its discounts with the members. Figure 16-1 shows a group owner providing discounts to three group members:

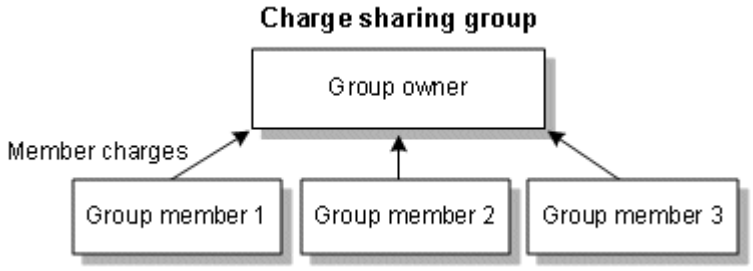
Figure 16-1 Discount Sharing Group



See "About Discount Sharing" for more information.

- **Charge sharing group:** The owner assumes charges that are incurred by the members. Figure 16-2 shows a group owner receiving charges from three group members:

Figure 16-2 Charge Sharing Group



Copyright © 2017, 2026, Oracle and/or its affiliates.

See "About Chargeshare Offers" for more information.

About First-Usage Start Time for Shared Balances

If a balance is shared among accounts in a sharing group, the balance validity period is set when any account in the group first impacts the balance. Because the same balance is shared with all accounts in the group, the validity period of that balance applies to all accounts.

About Discount Sharing

Discount sharing occurs when an account shares its discounts with other accounts. For example, a group of employees might share a pool of minutes in their company's mobile phone account, or a parent might purchase discounts on her email service and want those discounts to apply to her children's email services as well.

To share discounts in BRM, you create discount sharing groups. You set up the discounts in PDC.

BRM supports shared discounts by enabling a discount to apply either to the account that generates the event (the discount user), to the account that owns the discount, or to both.

With discount sharing, you sometimes need several discounts to specify how balances are granted or consumed for each account. For example, to offer 20% discount on usage to each member account when the total usage for all member accounts exceeds 1,000 minutes, you set up a discount with a counter to track the usage for each account and another discount to apply the percentage off based on the aggregated usage recorded in the counter balance.

About Chargeshare Offers

Charge sharing enables an account to sponsor the charges of other accounts. The owner account in a charge sharing group receives the balance impact of sponsored charges incurred by the member accounts. For example, charge sharing enables a company to pay for all of its employees' mobile phone services or a parent to pay for his child's SMS and email services.

To share charges in BRM, you create charge sharing groups. You then set up chargeshare offers and chargeshares in PDC to determine how charges are shared among the members of the charge sharing groups. If you configured product specification attributes for chargeshare offers in XML templates, you can set their values. See "Configuring Product Specification Attributes for Pricing Components".

A chargeshare uses rules that contain filters to determine whether an event qualifies for charge sharing, triggers to specify conditions that must be met before the chargeshare applies, and pricing to determine the charge sharing amounts and balance impacts.

Setting up chargeshare offers and chargeshares is similar to setting up discount offers and discounts. See "Creating Discount Offers" for more information.

Copyright © 2017, 2026, Oracle and/or its affiliates.

17

Creating Configurable Recurring Offers

Learn how to set up configurable recurring charge and discount offers in Oracle Communications Billing and Revenue Management (BRM), allowing customer service representatives to select billing attributes at the time of purchase.

Topics in this document:

- About Creating Configurable Recurring Offers That Can Be Changed At Purchase
- About the Product Specification Characteristic Attributes You Can Expose
- How Configurable Recurring Offers Work
- Loading the Product Specification Characteristic Templates
- Setting Up Configurable Recurring Offers at Purchase
- Configuring External Applications to Manage Product Spec Characteristic Attributes
- About Limitations When Using Configurable Recurring Offers

About Creating Configurable Recurring Offers That Can Be Changed At Purchase

You can create recurring charge offers and discount offers that CSRs can configure at purchase time. Instead of creating separate offers for every billing scenario, you expose a controlled set of attributes, such as cycle charge type, cycle frequency, and cycle alignment, for selection when the customer purchases the offer. A configurable product is an offer that contains a configurable recurring event. For example, if the customer purchases a configurable product and specifies they want a cycle forward charge type with a quarterly billing cycle, the CSR can set that at purchase.

The ability to override the attributes of an offer at purchase time eliminates the need for you to design multiple similar offers and helps streamline your product catalog.

> **Note**
>
> If you want to be able to change the rate associated with the configurable offer, you can add a rate plan selector to the offer as well.

About the Product Specification Characteristic Attributes You Can Expose

The templates shipped with PDC include the following attributes. In each offer, you must specify all attributes, and you can select one or more values for each attribute.

- **Cycle Charge Type:** Choose from **Cycle Forward**, **Cycle Arrear**, or **Cycle Forward Arrear**.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **Cycle Unit:** Set the unit to **Day(s)**, **Month(s)**, or **Year(s)**.
- **Cycle Frequency:** Specify a value greater than zero.
- **Cycle Alignment:** Select **Billing** or **Purchase**. Billing aligns the cycle start date to the customer's billing date while Purchase aligns it to the date of product purchase.
- **In Advance or In Arrear Billing Unit:** Choose **Day(s)** or **Month(s)**.
- **In Advance or In Arrear Billing Frequency:** Enter any negative or positive number.

How Configurable Recurring Offers Work

Configurable recurring offers use a configurable billing event named **/event/billing/product/fee/cycle/configurable**. BRM considers the base class to be **/event/billing/product/fee/cycle** and resolves the actual subclass at purchase time based on the customer's selection. For example, if a customer chooses a cycle forward charge type, BRM applies the **/event/billing/product/fee/cycle/cycle_forward** event. The two PDC product specification attribute templates (**PSC_ConfigurableEvent_ChargeOffer.xml** and **PSC_ConfigurableEvent_AlterationOffer.xml**) let you declare which attributes a CSR can set and which values are allowed. At purchase time, Billing Care (or your custom client application) presents only those attributes and values that are enabled in the offer. For example, if you allow the cycle unit to be Days or Months, Billing Care lets the CSR choose only Days or Months for that offer. When the CSR completes the purchase, BRM adds the selected values to the customer's purchased product and uses them for rating and billing going forward.

Loading the Product Specification Characteristic Templates

To load the attribute characteristic templates into PDC:

1. Go to *PDC_home*/**PDC/apps/bin**.
2. Run the following commands
   - For the charge offer template:
     ./ImportExportPricing -import -config ../Samples/Examples/PSC_ConfigurableEvent_ChargeOffer.xml -ow
   - For the discount offer template:
     ./ImportExportPricing -import -config ../Samples/Examples/PSC_ConfigurableEvent_AlterationOffer.xml -ow

Setting Up Configurable Recurring Offers at Purchase

To set up recurring offers that customers can configure at purchase:

- Ensure the **/config/event_map** object includes the cycle fee event subclasses you plan to support, such as:
  - **/event/billing/product/fee/cycle/cycle_forward**
  - **/event/billing/product/fee/cycle/cycle_arrear**
  - **/event/billing/product/fee/cycle/cycle_forward_arrear**
  - **/event/billing/product/fee/cycle/configurable**

Copyright © 2017, 2026, Oracle and/or its affiliates.

Note: Define the service-event mapping in BRM by using the **pin_usage_map** file and the **load_usage_map** utility. For more information, see "Mapping Event Types to RUMs" in *BRM Setting Up Pipeline Pricing*.

- Load the **PSC_ConfigurableEvent_ChargeOffer.xml** and **PSC_ConfigurableEvent_AlterationOffer.xml** templates into PDC using **ImportExportPricing** so that the configurable attributes are available in the PDC UI and XML pricing interface. For more information, see "Loading the Product Specification Characteristic Templates".
- When you create charge and discount offers in PDC:
  - Select **Subscription** as the type and select the recurring event type **/event/billing/product/fee/cycle/configurable** as the event type for the recurring fee.
  - Under **Product Spec Characteristic Attributes**, specify the allowed values for each attribute.
  - If needed, add a rate plan selector.

To create configurable offers in **ImportExportPricing**, see "Configuring Product Specification Attributes for Pricing Components".

- If you are using a custom client application, customize it to call the Billing Care REST API or the BRM Subscription opcodes. See "Configuring External Applications to Manage Product Spec Characteristic Attributes".
- When you create an account in Billing Care or a custom client application, the CSR selects the cycle attributes to apply to the offer. See "Creating an Account" in *Billing Care Online Help.*

Configuring External Applications to Manage Product Spec Characteristic Attributes

If you want to use an external application to manage product spec characteristic attributes, you must customize it to pass product spec characteristic attributes when purchasing charge offers and discount offers. You can do this using one of these methods:

- The PCM_OP_SUBSCRIPTION_PURCHASE_PRODUCT opcode when purchasing charge offers, and the PCM_OP_SUBSCRIPTION_PURCHASE_DISCOUNT opcode when purchasing discount offers. Ensure you pass the PIN_FLD_PRODUCT_CHARACTERISTIC array with the set of allowed attributes and values. See "Specifying Cycle and Billing Attributes" in *BRM Opcode Guide*.
- Using the Billing Care REST API endpoints listed below. For each endpoint, ensure you include the following:

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 17-1 Endpoint for Specifying Cycling and Billing Attributes

API Endpoint	Request Payload
Create an Account (/bcws/webresources/v1.0/accounts)	Use the following request payload fields under **customizedChargeOffers** or **customizedDiscountOffers**: – cycleChargeType – cycleUnit – cycleFrequency – cycleAlignment – inAdvanceBillingUnit – inAdvanceBillingFrequency
Add Service to an Account (/bcws/webresources/v1.0/accounts/{id}/package)	Use the following request payload fields under **customizedChargeOffers** or **customizedDiscountOffers**: – cycleChargeType – cycleUnit – cycleFrequency – cycleAlignment – inAdvanceBillingUnit – inAdvanceBillingFrequency
Add bundle to a service (/bcws/webresources/v1.0/accounts/{id}/bundle)	Use the following request payload fields under **customizedChargeOffers** or **customizedDiscountOffers**: – cycleChargeType – cycleUnit – cycleFrequency – cycleAlignment – inAdvanceBillingUnit – inAdvanceBillingFrequency

About Limitations When Using Configurable Recurring Offers

Be aware of the following limitations when creating configurable recurring offers:

- After customers purchase a configurable offer and set the configurable attributes, they cannot be modified again.
- When you create an offer in PDC, it can contain only one configurable recurring offer or one standard recurring offer. It can include other regular offers, but only one recurring offer of either type.
- The configurable recurring offer has one fee, but your customers can choose the other attributes.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Part III
Charging Scenarios

This part describes various charging scenarios you can configure using Oracle Communications Billing and Revenue Management (BRM) pricing components and setup components in Pricing Design Center (PDC).

This part contains the following chapters:

- Charging Based on Date and Time
- Charging Based on Event Attributes
- Implementing Recurring Charges
- Advanced Configuration for Recurring Charges
- Working with Extended Rating Attributes
- Configuring Closed User Groups

Copyright © 2017, 2026, Oracle and/or its affiliates.

18

Charging Based on Date and Time

Learn how to charge for usage events based on date and time in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

- About Charging Based on Date and Time
- About Special Day Calendars
- Rating Events Split Across Time Periods

About Charging Based on Date and Time

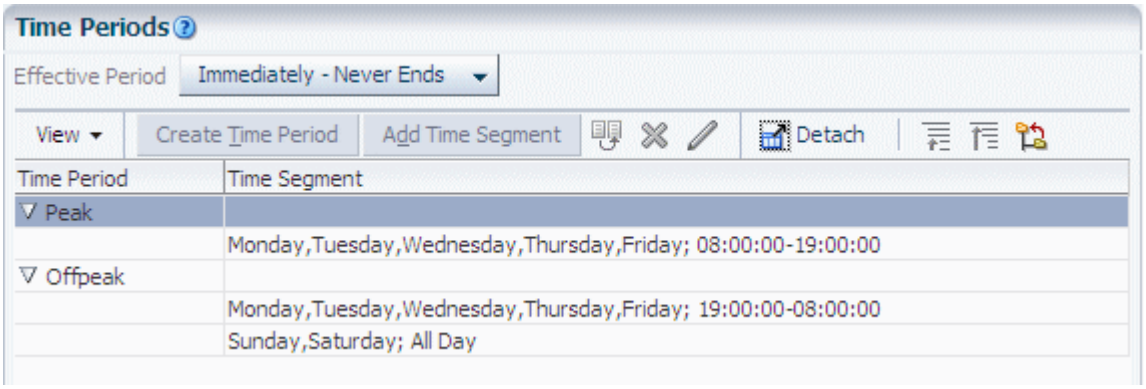
You can configure charge offer pricing to charge different amounts depending on the date and time. For example, you can create different pricing for peak and off-peak times. To do so, you use time models.

A time model is a set of time periods. Each time period contains one or more time segments. A time segment represents a particular time, such as a day of the week or a range of several hours.

By default, a time period is immediately and forever effective. To add effective periods to a time period, you specify only the start date of the new period. That date becomes the end date of the previous period. The end date of the final effective period is always **Forever**.

You can define time segments by using months, days of the month, days of the week, time of day, and a calendar of special days. For example, Figure 18-1 shows an off-peak and a peak time period whose time segments are defined by days of the week and time of day.

Figure 18-1 Peak and Off-Peak Time Periods in a Time Model



About Special Day Calendars

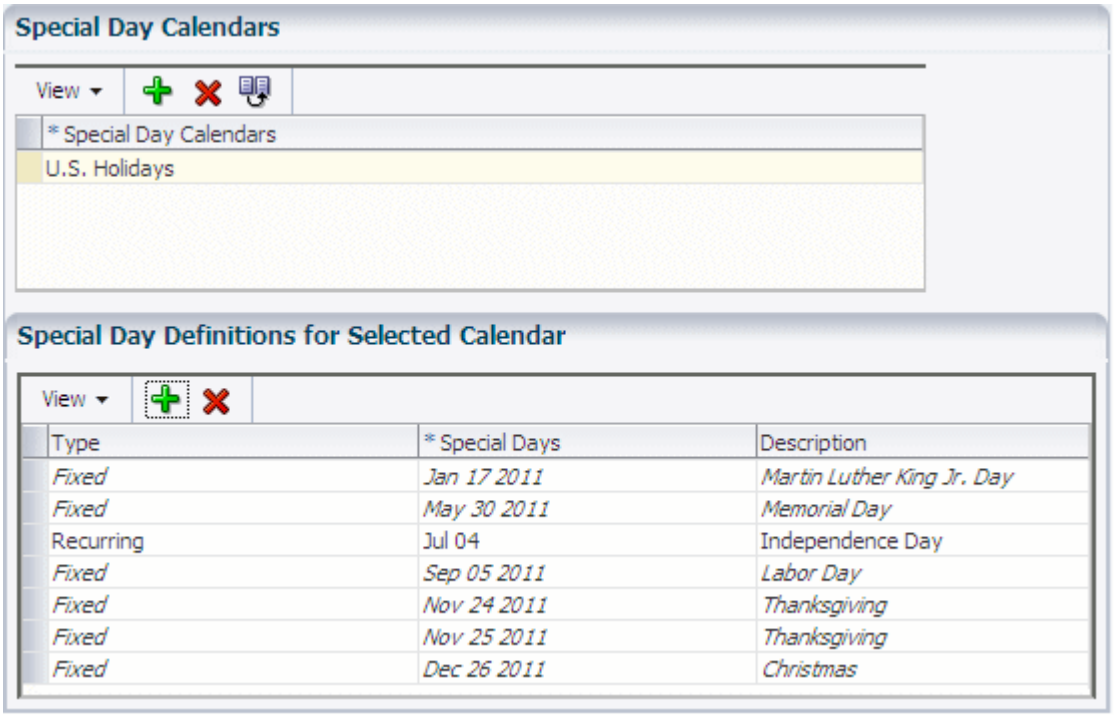
A special day calendar is a set of dates, such as holidays, for which you want to charge special prices for your services. Each date must be one of the following types:

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **Fixed:** A specific date valid only in *one year,* such as May 8, 2011, for Mother's Day in the U.S.
- **Recurring:** A date that is valid *every year,* such as July 4 for Independence Day in the U.S.

Figure 18-2 shows a special day calendar that has fixed and recurring dates.

Figure 18-2 Special Day Calendar



To configure pricing for special days in a charge, you associate a special day calendar with a time model. You must then configure at least one time period that applies to the special days. The time model should cover all 24 hours of the special days.

> **Note**
>
> The same time period cannot apply to both regular days of the week and special days. If you configure a time period that applies to both, the time model receives a validation error.

The same calendar can be associated with multiple time models, but you can also create different calendars for different time models.

For more information, see "Creating Special Day Calendars" in *PDC Online Help*.

Rating Events Split Across Time Periods

Events sometimes overlap time periods. For example, if off-peak rating starts at 7:30 p.m., and a call begins at 7:10 p.m. and ends at 7:35 p.m., the call overlaps the boundary between peak and off-peak rates.

Copyright © 2017, 2026, Oracle and/or its affiliates.

ECE uses the same rating scope for candidate selection and final rating. When ECE splits an event across time periods, it evaluates the candidates separately for each rating cut. As a result, different rating cuts can select different Charge Selector rules or Generic Selector result branches.

When an event crosses multiple time periods, you can specify the way to charge for an event if it is priced differently by time period:

- **Start Time:** Applies the charges defined in the time period in which the event begins.
- **End Time:** Applies the charges defined in the time period in which the event ends.
- **Split:** Applies charges according to the time period in which each portion of the event falls.

If you split charges across time periods, and you configure pricing based on quantity, you can specify the way a price step is selected when part of the event was already charged in another time period.

- **Dependent On:** Dependent on the quantity already used. When selecting a price step, BRM considers the amount of the event that occurred in the previous time period.
- **Independent Of:** Independent of the quantity already used. When selecting a price step, BRM does not consider the amount of the event that occurred in the previous time period.

For example, you might create pricing such as:

- Off-peak pricing between 8 PM and 6 AM.
- During peak time, charge 25 cents per minute.
- During off-peak time, charge:
  - $0.10 per minute for the first hour
  - $0.05 per minute for subsequent hours

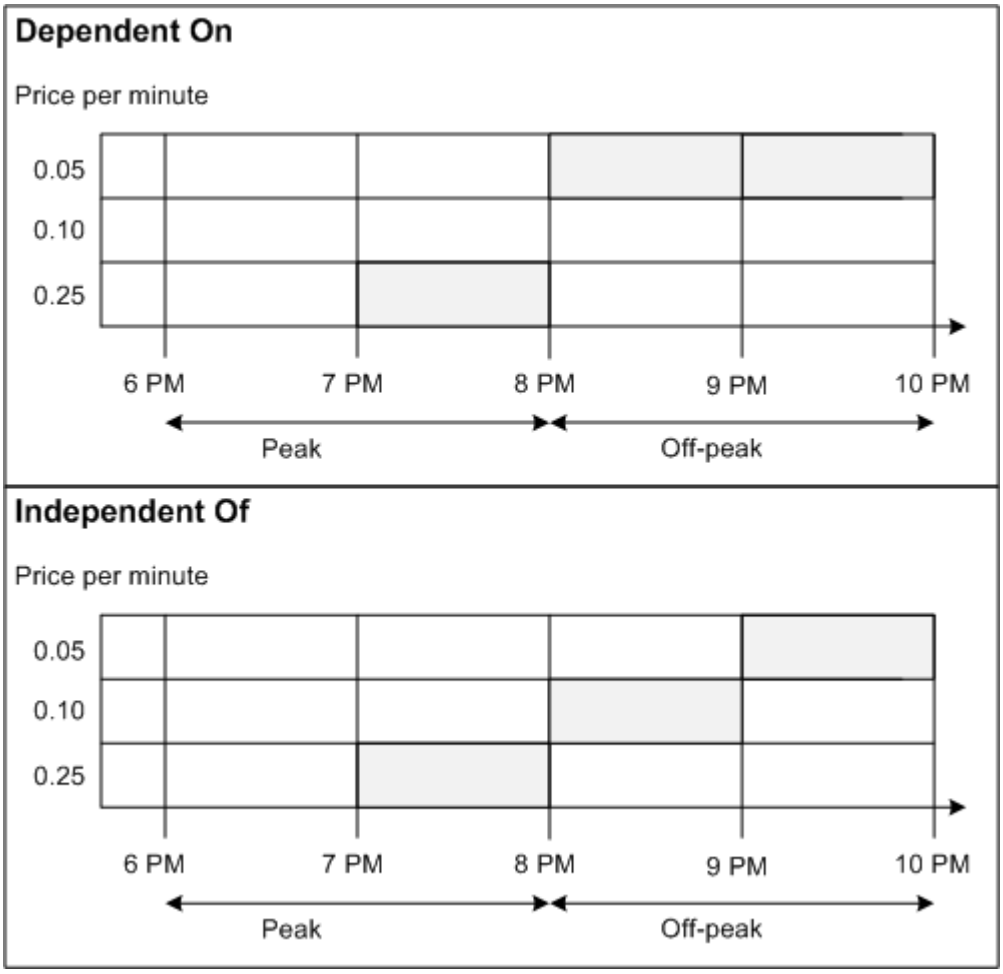
A phone call starts at 7 p.m. and ends at 10 p.m. This results in one hour of peak time, and two hours of off-peak time.

- If the selection quantity is **Dependent On** the quantity already used, the hour charged as peak time is considered as the first hour, and the entire off-peak portion of the call is charged $0.05 per minute.
- If the selection quantity is **Independent Of** the quantity already used, the hour charged as peak time is **not** considered as the first hour, and the off-peak portion of the call is charged $0.10 per minute for the first hour, and $0.05 for the second hour.

Figure 18-3 shows how the call is charged.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Figure 18-3 Quantity-Based Pricing Split by Time Periods



Copyright © 2017, 2026, Oracle and/or its affiliates.

19

Charging Based on Event Attributes

Learn how to charge for usage events based on event attributes in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

- About Impact Categories
- About Zone Models
- About Value Maps
- About Selectors
- About Usage Scenario (USC) Selectors
- About Mapping Services

About Impact Categories

You use impact categories to enable the same charge to apply different pricing based on the values of various event attributes. For example, to configure different pricing for calls made to different countries, you add impact categories for each destination country to the charge. When a call occurs, the pricing associated with the impact category for that call's destination is applied to the call. For example, to charge usage for a mobile phone service, you might create the following impact categories:

- Impact Category: **Bolivia**

  Balance Impact: **.10 per minute**
- Impact Category: **France**

  Balance Impact: **.05 per minute**

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. For example, you can create impact categories to rate:

- Local calls
- Long distance calls
- Local call forwarding
- Long distance call forwarding

Impact categories for zones are used as follows:

- *Base* zone impact categories are used as the results of rules in zone models and in Usage Scenario (USC) selectors.
- *Derived* zone impact categories are used as the results of rules in USC selectors and Access Point Name (APN) selectors. They are considered derived because the rules in USC and APN selectors use additional attributes to remap base zone impact categories to different zone impact categories. For example, a USC selector rule might specify that if the

Copyright © 2017, 2026, Oracle and/or its affiliates.

base zone impact category is China and the usage type is **Friends & Family**, the selector returns the Friends & Family derived zone impact category.

You can group multiple impact categories into a single group node to apply the same pricing to all of them.

About Zone Models

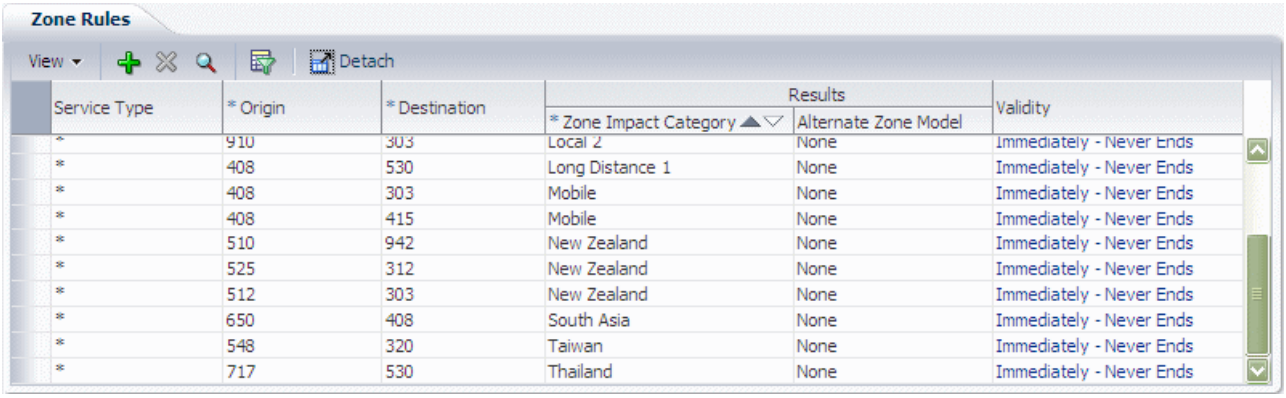
You use zone models to charge for calls based on their origin and destination. BRM supports the following types of zone models:

- **Standard:** A zone model based on the origin and destination numbers of a call. It contains rules that associate a pair of origin and destination numbers with a zone impact category (see "About Impact Categories").

  To specify origin and destination numbers, you must include an international access code (the code used to dial *out of* the country in which the phone number is located). Optionally, you can also include a country code (the code used to dial *in to* the country in which the phone number is located), an area code, a region code, a city code, a phone number prefix, and so on—up to and including the entire phone number.

  Figure 19-1 shows zone rules associated with the same impact category.

Figure 19-1 Multiple Zone Rules Associated with the Same Impact Category



- **Geographical:** A zone model based on the distance between the origin and the destination of a call. Geographical zone models include the following:
  - Zone rules that associate a distance with a zone impact category.
  - A list of area codes. Each area code is associated with one or more pairs of longitude and latitude coordinates.

  When a customer makes a call, BRM uses the data in the area code list to compute the distance between the origin number and the destination number. BRM then assigns an impact category to the call event by using the zone rule whose distance most closely matches the computed distance.

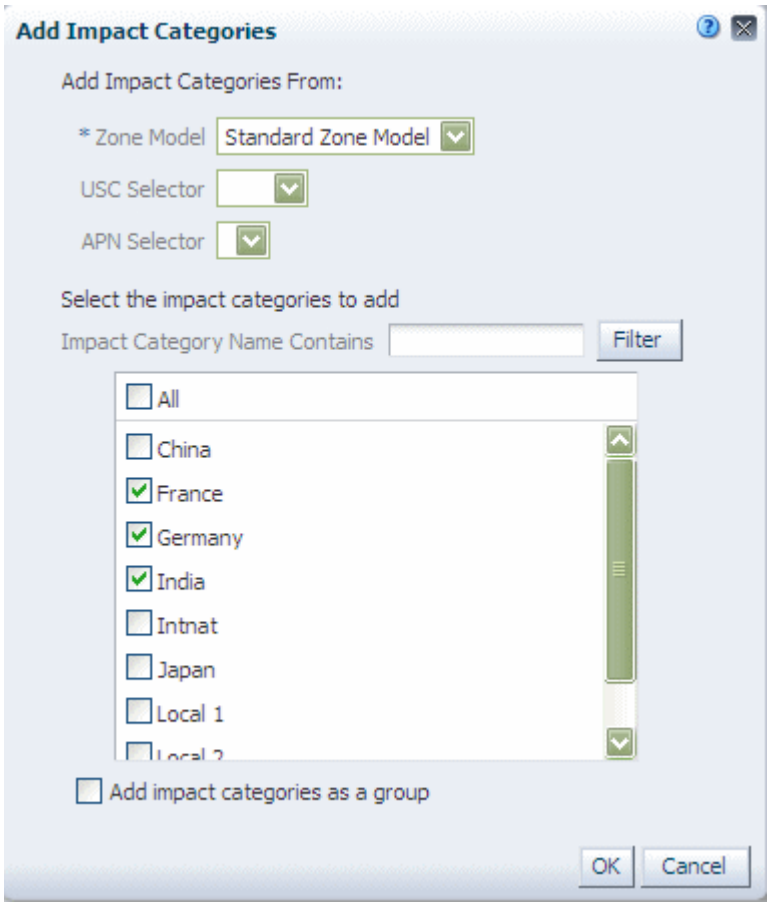
  Geographical zone models are useful in the following situations:

  - Customers are located close to 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 associating different pairs of latitude and longitude coordinates with the same area code in a geographical zone model.

Copyright © 2017, 2026, Oracle and/or its affiliates.

When creating charges, you must select a zone model to get a list of impact categories that can be added to the charge. For example, Figure 19-2 shows how the selected zone model determines which impact categories are available.

Figure 19-2 Using Zone Models to Select Impact Categories



> **Note**
>
> Optionally, you can also select a USC or APN selector to provide enhanced zone impact categories. See "About Selectors" and "About Impact Categories" for more information.

Alternate Zone Models

In both standard and geographical zone models, each rule results in the application of a zone impact category and, optionally, an alternate zone model (see the rules in Figure 19-1). Alternate zone models enable you to define more granular zone rules and reuse them in multiple zone models.

For example, in a zone model for calls that originate in the U.S., you might include a rule with the following parameters:

- **Origin:** 011

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **Destination:** 00
- **Impact Category:** 00 General
- **Alternate Zone Model:** US to 00 Country Codes

In this example, the alternate zone model might contain rules for various combinations of calls from the U.S. (011 international access code) to numbers with the 00 international access code and a particular country code, such as **00 44** for calls to the U.K., **00 91** for calls to India, and so on.

When a call is made from the U.S. to a 00 international access code in this example, BRM first checks whether the country code of the destination number matches a rule in the alternate zone model. If it does, BRM uses the impact category associated with the rule in the alternate zone model to calculate the call's charge. If it does not, BRM uses the 00 General impact category to calculate the charge.

See "Creating Pricing Setup Components" for more information about configuring zone models.

Displaying Zoning Information on an Invoice

You can enter a zone description that can be included in invoices. To do so, enter the zone description in the **Description** field in the following dialog boxes:

- Standard Zone
- Geographical Zone
- Usage Scenario Mapping

You can use a maximum of 2,000 characters in the description.

About Value Maps

A value map is a hierarchical structure that associates zone names with values that apply to a single event attribute, such as an area code, an IP address, or a broadband bit rate. A value map can have multiple nested zones.

You use value maps in charge selectors to group event attribute values into manageable categories. For example, to apply the same charge to all calls made from the San Francisco Bay Area to Los Angeles, you might use a California Area Codes value map that includes the following zones:

- **Northern California**
  - **San Francisco Bay Area**
    - 408
    - 415
    - 510
    - 650
    - 707
- **Southern California**
  - **Los Angeles**
    - 213
    - 323

Copyright © 2017, 2026, Oracle and/or its affiliates.

Instead of creating charge selector rules for all possible combinations of those area codes, you associate the California Area Codes value map with the charge selector and then create only one rule in which **San Francisco Bay Area** is the origin call value and **Los Angeles** is the destination call value.

> **Note**
>
> In PDC, value maps are used only with charge selectors (see "About Selectors").

See "Creating Pricing Setup Components" for information about configuring value maps.

About Selectors

You use selectors to apply different charges, discounts, prices, tax codes, or tax exemption codes to the same event based on the value of one or more service, event, or customer attributes. For example, you could charge different fees for phone calls depending on a call's origin and destination.

A selector assigns a result tag to events that have specific attributes. For example, it could assign the **SFtoNYC** result tag to calls made from San Francisco to New York City, and the **SFtoCHC** result tag to calls made from San Francisco to Chicago. Later, when you add the selector to a charge offer, you can map one or more result tags to a specific charge, discount, price, and so on.

You can create the following types of selectors:

- **Charge selector:** Determines the charge to use based on specified event, service, or account attributes. You can use a charge selector instead of a charge in a charge offer.
- **Discount selector:** Determines the discount to use based on specified event, service, or account attributes. You can use a discount selector instead of a discount in a discount offer.
- **Generic selector:** Determines the result tag based on event attributes, service attributes, or custom rules that are used in a charge offer to guide pricing.
- **Price selector:** Determines the appropriate pricing to use based on specified event, service, or account attributes. You can use a price selector instead of a pricing instance in a charge offer.
- **Usage Scenario (USC) selector:** Determines a new impact category based on the zone impact category and specific event attributes.
- **Access Point Name (APN) selector:** Determines a new impact category based on the zone impact category and access point name that applies to the event.
- **Tax selector:** Determines the tax code to apply based on service, event, account, or profile attributes.
- **Tax Exemption selector:** Determines the tax exemption code to apply based on service, event, account, or profile attributes.

When creating a selector, you create rules and choose the attributes to use in them. Each rule uses the same fields, and you must specify a value for every field in every rule. For fields that are not relevant to a rule, use a wildcard (*) to indicate that any value is acceptable.

The execution sequence of the rules is important, so you can reorder them as necessary. The rules tables include a search mechanism to make it easy to find a rule.

Copyright © 2017, 2026, Oracle and/or its affiliates.

In all selectors except charge selectors, rules have an effective period. By default, the period starts immediately and never ends. You can modify and add effective periods.

You can create selectors using:

- PDC UI. See *PDC Online Help* for more information.
- The **ImportExportPricing** utility. See "XML Examples of Creating Pricing Components" for more information.

Impact-Based Candidate Selection

By default, a charge selector selects the first matching rule in configured order, and a generic selector selects the first matching result branch. You can instead configure these selectors to use impact-based candidate selection when multiple candidates can match the same event.

Impact-based candidate selection applies only to charge selectors and generic selectors during Elastic Charging Engine (ECE) rating. Impact-based candidate selection enables ECE to evaluate all matching candidates and select a candidate based on the actual rating impact to a specified balance element. You might use this feature when overlapping selector rules can produce different charges and you want ECE to select the candidate that produces the lowest or highest impact rather than selecting whichever candidate happens to match first.

> **Note**
>
> Impact-based candidate selection applies only to charge selectors and generic selectors and is used only by ECE for rating. For more information, see "How Impact-Based Candidate Selection Works" in *ECE Implementing Charging*.

Impact-based candidate selection compares the actual rating impacts produced during ECE rating, not the rate values configured for the candidates. Therefore, MIN_IMPACT does not necessarily select the candidate with the lowest configured rate, and MAX_IMPACT does not necessarily select the candidate with the highest configured rate. For example, a candidate with a higher configured rate can produce a lower rating impact when rating consumes free units, crosses tiers, or produces multiple balance impacts.

You configure candidate selection by specifying Candidate Selection Mode for the charge selector or generic selector. The following modes are available:

- First Match (**FIRST_MATCH**) selects the first matching charge selector rule or generic selector result branch in configured order.
- Lowest Value (**MIN_IMPACT**) evaluates all matching candidates and selects the candidate with the lowest signed immediate impact for the configured balance element.
- Highest Value: **MAX_IMPACT** evaluates all matching candidates and selects the candidate with the highest signed immediate impact for the configured balance element.

For Lowest Value and Highest Value, you must also specify Candidate Selection Balance Element Code, which identifies the currency or noncurrency balance element whose impact ECE uses to select the candidate.

Existing selectors that do not specify **CandidateSelectionMode** continue to use **FIRST_MATCH**.

You can configure candidate selection by using one of the following:

Copyright © 2017, 2026, Oracle and/or its affiliates.

- The PDC UI. To do so, use the **Candidate Selection Mode** and **Candidate Selection Balance Element Code** fields. See "Specifying Charge Selector General Information" and "Specifying Generic Selector General Information" in *PDC Online Help*.
- The **ImportExportPricing** utility. To do so, in your input XML file, set the **<CandidateSelectionMode>** and **<CandidateSelectionBalanceElementCode>** elements under **<AttributeAnalyzerModel>**. See "Configuring Impact-Based Candidate Selection by Using the XML Pricing Interface".

If the candidate selection mode is not specified, the selector uses **FIRST_MATCH**, preserving the behavior of existing selector configurations.

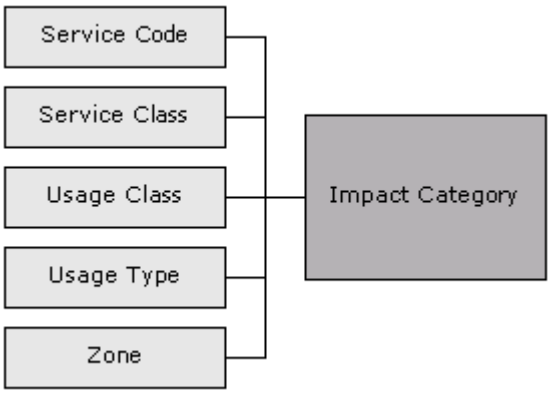
About Usage Scenario (USC) Selectors

A Usage Scenario (USC) selector contains rules that map existing impact categories to new impact categories based on usage attributes. For example, if you provide a friends-and-family discount, you can create separate impact categories for national friends and international friends.

You define USC selector rules by specifying values for the existing impact category, the event usage attributes used to evaluate the USC rule, and the new impact category. When an event's usage attribute values match a rule, the new impact category is returned by the USC selector.

USC selectors bring together the data from service codes, usage classes, and zones, as shown in Figure 19-3.

Figure 19-3 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.

You can use multiple attributes to define an impact category, or you can use only one attribute.

About Mapping Services

Incoming events from multiple switches often use different codes to represent the same service or supplementary service. To process these events, you must normalize that data by mapping external service codes to internal service codes, service classes, and usage classes.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- 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 mailbox). The data used for rating usage classes comes from the network.

To create internal service mappings, you 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**.

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. You can create any number of map groups.

Mapping Service Codes and Service Classes

To create service code and service class mappings, you 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 **/service/telco/gsm/telephony**.

The mappings are based on the following data:

- External service code
- Usage class
- Location area indicator
- VAS event product code
- Quality of service requested
- Quality of service used
- Record type

Multiple 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 service codes to a single **/service/telco/gsm** BRM service type.

Mapping Usage Classes

To create usage class mappings, you 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 event 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.
- Service class mapping: Use the usage class to create special sub-services or quality of services (for example, service agreements).
- Usage scenario mapping: Use the usage class to map to impact categories.
- Pricing adjustment: Use the usage class to provide discounts or adjustments in individual events.

Copyright © 2017, 2026, Oracle and/or its affiliates.

The mappings are based on the following data:

- External usage class
- Tariff class and subclass
- Record type
- Connect type and Connect subtype, for example, anonymous, from another network, or direct
- Transit area code
- APN address when you use GPRS or UMTS technology
- SS (supplementary service) packet

Copyright © 2017, 2026, Oracle and/or its affiliates.

20

Implementing Recurring Charges

Learn how to implement and manage recurring charges in Pricing Design Center (PDC) for Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

- About Recurring Charges
- About Cycle Forward Charges
- About Cycle Arrears Charges
- About Cycle Forward Arrears Charges
- About Creating Recurring Charges in PDC
- About Recurring Charges Events

See also "Advanced Configuration for Recurring Charges".

About Recurring Charges

Recurring charges are ongoing charges that are not generated or affected by usage, such as a charge for a monthly subscription fee. For example:

- A 30-dollar monthly fee for a mobile phone service
- A 5-dollar fee for an automatically renewing smartphone app

Recurring charges do not use Elastic Charging Engine (ECE) for rating. Instead, recurring charges are applied when you run BRM billing utilities.

There are three types of recurring charges:

- A *cycle forward charge* applies a recurring charge in advance. The advantage of using a cycle forward charge is that you collect payments sooner.
- A *cycle arrears charge* applies a recurring charge to the previous month. The advantage of using a cycle arrears charge is that all of the usage charges and recurring charges for a particular month are included in the same bill.
- A *cycle forward arrears charge* applies a recurring charge to the previous month. identical to a cycle arrears charge. The advantage of using a cycle forward arrears charge is that the customer is not charged until the end of the month, but the balance impact is made at the beginning of the month, and can be recognized as unearned revenue at the beginning of the month.

About Cycle Forward Charges

To charge a fee in advance, create a charge offer that includes a *cycle forward* charge type, for example, **Monthly Recurring Charge**. You can charge fees in advance at any interval; for example, by using a **Quarterly Recurring Charge** or a **Yearly Recurring Charge** charge type. You can define your own recurring charge types; for example, weekly, or every five months (see "Configuring Flexible Cycle Forward Charges" for more information).

Copyright © 2017, 2026, Oracle and/or its affiliates.

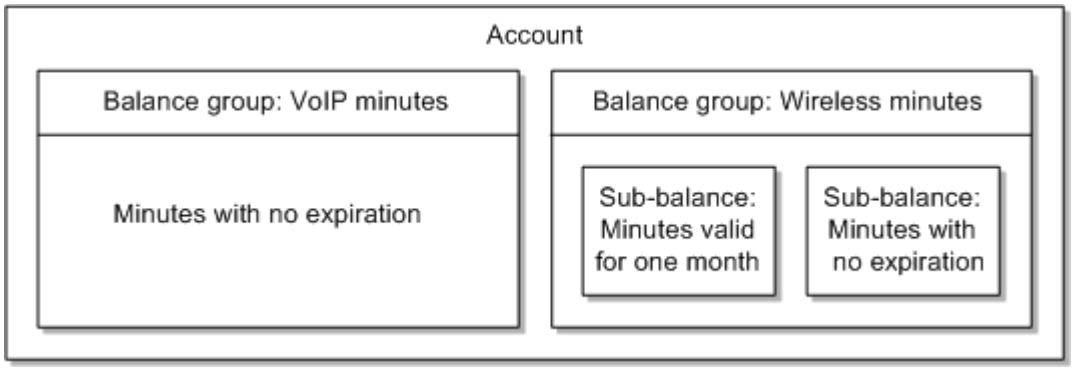
Monthly recurring charges are typically applied at the beginning of an accounting cycle, but they do not need to be synchronized to the start or end of an accounting cycle.

Cycle forward balance impacts are recorded in **/item/cycle_forward** bill items.

About the First Cycle Forward Charge in a New Account

A cycle forward fee incurred at account creation is included in the cycle forward item created when the account is created. Therefore, the first bill includes two cycle forward fees. Figure 20-1 shows an account created on May 15, with the first bill on June 15. The first cycle forward item includes the charge for the month from May 15 - June 15. The second cycle forward item includes the charge for June 15 - July 15. Both charges are included in the June 15 bill.

Figure 20-1 Cycle Forward Fees for New Account



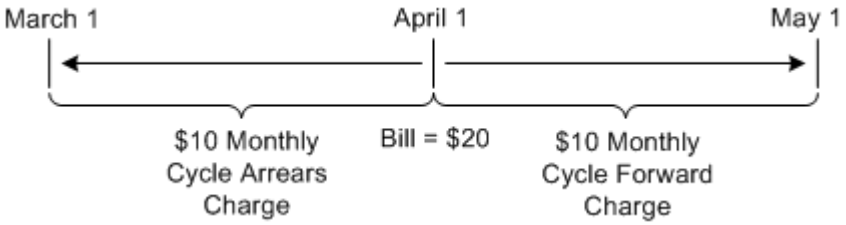
About Cycle Arrears Charges

To charge a fee at the end of the month, create a charge offer that includes a *cycle arrears* charge type, for example, **Monthly Recurring Charge Arrear**. Cycle arrears charges apply a charge to the previous billing cycle; the customer pays for the month that has already occurred. Cycle arrears charges are always monthly.

In Figure 20-2, the customer is charged for a cycle forward fee and for a cycle arrears fee. The bill created on April 1 includes:

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

Figure 20-2 Cycle Forward Charge and Cycle Arrears Charge



Copyright © 2017, 2026, Oracle and/or its affiliates.

Cycle arrears balance impacts are recorded in **/item/cycle_arrear** bill items.

About Cycle Forward Arrears Charges

To charge a fee at the end of the month, and recognize unbilled revenue when you run general ledger reports, create a charge offer that includes a *cycle forward arrears* charge type, for example, **Monthly Recurring Charge Forward Arrears**.

Just like a cycle arrears charge, a cycle forward arrears charge charges the customer for the previous billing cycle. The difference between cycle arrears charges and cycle forward arrears charges is:

- The balance impact for a *cycle arrears charge* is applied at the **end** of the accounting cycle.
- The balance impact for a *cycle forward arrears charge* is applied at the **beginning** of the accounting cycle.

However, for both types of charges, the charge is not billed until the end of the billing cycle. Therefore, to the customer, there is no difference in how the charge is billed. The difference appears to your accounting department when you collect general ledger data. Cycle forward arrears charge types allow you to recognize unbilled revenue for the charge when you collect general ledger data before the end of the accounting cycle.

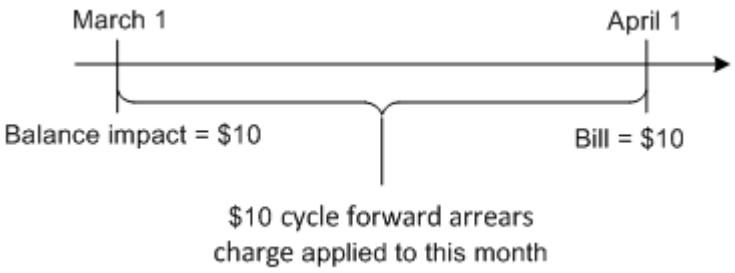
Cycle forward arrears charges support multi-month intervals. You can configure cycle forward arrears charges to apply for multiple months, quarters, or other defined periods.

> **Note**
>
> For cycle forward arrears charges configured with an advance billing offset, BRM adds future-cycle charges to the next bill. For example, if the billing period is quarterly, the recurring charge cycle is monthly, and the billing offset is three months, the monthly charges for the upcoming quarter appear on the following bill instead of the current bill.

Cycle forward arrears balance impacts are recorded in **/item/cycle_forward_arrear** bill items. However, the cycle forward arrears event is assigned to the 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 the cycle.

Figure 20-3 shows a cycle forward arrears charge applied on March 1, and billed on April 1. In this case, the balance impact occurs on March 1, but the charge is included in the bill item for April 1.

Figure 20-3 Cycle Forward Arrears Charge



Copyright © 2017, 2026, Oracle and/or its affiliates.

About Implementing Cycle Forward Arrears Charging

Cycle forward arrears events are standard in BRM. However, you must enable delayed billing if you charge for cycle forward arrears events. If you do not otherwise need delayed billing, you can enable delayed billing, but set the delayed billing period to **0**.

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

About Creating Recurring Charges in PDC

You create a recurring charge in PDC by assigning a recurring charge type to a charge offer. You then apply the same attributes as any charge, for example:

- How the event is measured. Recurring charges are always measured by occurrence.
- When to stop charging; for example, when the associated charge offer is inactivated or closed.
- Tax calculation settings.
- General Ledger (G/L) impacts. You can configure general ledger (G/L) reporting to report earned revenue and unearned revenue. Unearned revenue applies only to cycle forward events and cycle forward arrears events. It never applies to usage events or cycle arrears events.
- Balance impact details, such as the amount to charge, balance element, and so on.

In addition, you can specify when the balance impact of a recurring charge is applied, and whether to prorate recurring charges. See:

- Specifying When Recurring Charges are Charged
- Allowing Recurring Charges to Be Prorated

> **Note**
>
> A single charge offer cannot include multiple recurring charges 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 charge offer, you cannot also add a monthly cycle arrears or monthly cycle forward arrears event to the same charge offer. Instead, Oracle recommends that you create separate charge offers for each cycle forward event.

Specifying When Recurring Charges are Charged

In PDC, you can configure the balance impact for recurring charges to occur at the following times:

- On the customer's current billing date. This is the default. If the recurring charge is in a charge offer that is purchased mid-cycle, you can prorate a charge to cover the time between the purchase date and the first billing date.
  For example, if the billing date is the 1st of the month and the charge offer is purchased on January 10, the charge is prorated and applied on January 10 for the interval January 10 to

Copyright © 2017, 2026, Oracle and/or its affiliates.

February 1. For subsequent cycles, the charge is applied on the billing date (for the interval February 1 to March 1, March 1 to April 1, and so on).

- On the charge offer purchase date. For example, if the billing date is the 1st of the month and the charge offer is purchased on January 10, the charge is applied on the 10th of every month (for the interval January 10 to February 10, February 10 to March 10, and so on).
- At a specified time before the time period covered by the charge; for example, one month in advance. For example:
  - If you charge a one-month cycle forward fee one month in advance, and the billing day of month (DOM) is the 1st, the charge on May 1st applies to June 1 - June 30. Without in-advance billing, a cycle forward charge on May 1 applies to May 1 - May 31.
  - If you charge a one-month cycle forward fee 15 days in advance, and the billing day of month (DOM) is the 1st, the charge on May 1 applies to May 15 - June 15, the charge on June 1 applies to June 15 - July 15, and so on.

  When you charge cycle forward fees in advance, the first cycle forward charge might be larger than the subsequent charges. For example, if a charge offer is purchased on May 1 with in-advance billing of 15 days, the charge on May 1 includes one month in advance (May 15 - June 15) plus 15 days (May 1 - May 14). Subsequent charges are for June 15 - July 15, and so on.

If you create pricing components using the **ImportExportPricing** utility, you can also apply recurring charges on a specific day of the month. For example, you can apply the recurring charge on the 5th day of each month regardless of the billing or purchase date. This is useful for offers with validity that is defined in hours, minutes, and seconds rather than days, weeks, and months. See "Aligning Recurring Charges and Product Validity to a Specific Day of the Month" for more information.

Allowing Recurring Charges to Be Prorated

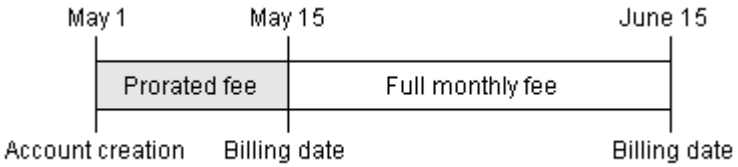
You can enable recurring charges to be prorated. A prorated charge typically charges a portion of the fee, based on how long it has been applied. You can also specify to apply no fee, or the full amount. You can specify to prorate charges in the following cases:

- When a charge offer is purchased mid-cycle. This can occur at account creation or when a charge offer is purchased by an existing account.
- When a charge offer that includes a recurring charge is canceled mid-cycle.

If a recurring charge includes more than one balance impact, you can specify which balance impacts should be prorated. For example, if the charge can be prorated, and there are two balance impacts, you can specify to prorate one of the balance impacts but not the other.

Figure 20-4 shows a cycle forward charge applied at account creation on May 1. The billing day of month is May 15, so the cycle forward charge is prorated for the two weeks between account creation and the first billing date.

Figure 20-4 Prorating Cycle Charges for a Monthly Fee



Copyright © 2017, 2026, Oracle and/or its affiliates.

About Recurring Charges Events

Like all charges, recurring charges are rated by applying a balance impact to an event. Recurring charges are applied to *cycle events*. The cycle events are:

- **/event/billing/product/fee/cycle/cycle_forward_monthly**
- **/event/billing/product/fee/cycle/cycle_forward_bimonthly**
- **/event/billing/product/fee/cycle/cycle_forward_quarterly**
- **/event/billing/product/fee/cycle/cycle_forward_semiannual**
- **/event/billing/product/fee/cycle/cycle_forward_annual**
- **/event/billing/product/fee/cycle/cycle_arrear**
- **/event/billing/product/fee/cycle/cycle_forward_arrear**

Cycle events are created when you run the **pin_cycle_fees** billing utility. The **pin_cycle_fees** utility is typically included in the **pin_bill_day** script that runs daily.

About Configurable Recurring Charges

Configurable recurring charges allow you to override billing cycle settings such as cycle charge type, cycle unit, and cycle frequency at the time you purchase a charge or discount offer. For example:

- A monthly data plan in which the customer selects whether to pay in advance or in arrears.
- A yearly subscription where the provider decides at purchase time whether to align charges to the bill cycle or the date of purchase.
- A quarterly recurring fee that you configure as part of a charge offer.

With configurable recurring charges, you do not need to create multiple offers for every billing configuration. This reduces the need to create a separate charge or discount offer for each kind of configuration.

You can define cycle and billing attributes at design time in PDC. See “Setting Cycle and Billing Attributes for Charge Offer” and “Setting Cycle and Billing Attributes for Discount Offer” for more information. You can set these attributes at purchase time in Billing Care, see "Creating an Account" in *Billing Care Online Help* for more information.

Copyright © 2017, 2026, Oracle and/or its affiliates.

21

Advanced Configuration for Recurring Charges

Learn how to configure advanced functionality for recurring charges in Pricing Design Center (PDC) for Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

- Configuring Flexible Cycle Forward Charges
- Applying Cycle Forward Fees in Parallel
- Applying Recurring Charges Based on Charge Offer Priority
- Customizing the Cycle Interval for Specified Charge Offers

See also "Implementing Recurring Charges".

Configuring Flexible Cycle Forward Charges

By default, BRM supports monthly, bimonthly, quarterly, semi-annual, and annual cycle forward charges. You can also configure BRM to support *flexible cycle forward charges*. Flexible cycle forward charges can use daily, weekly, monthly, or multimonth cycles that are not restricted to billing or accounting cycles.

You can use flexible cycle forward charges to grant free balances, provide discounts, or charge fees at any time during the accounting cycle. For example, you can set up a cycle forward fee to grant minutes every week or every day rather than once a month. Or you can set up a monthly cycle forward fee to grant minutes on the 15th of every month, which differs from the monthly accounting cycle that begins on the 1st of every month.

Because flexible cycles are not aligned with accounting cycles, cycle forward fees are always prorated.

You set up flexible cycle forward charges by configuring flexible cycle forward events.

To set up flexible cycles:

1. Define a custom cycle forward event subclass using Storable Class Editor in Developer Center. The name of the custom cycle forward event must use this syntax:

   `/event/billing/product/fee/cycle/cycle_forward_`*`customName`*

   where *customName* is your custom name for the event subclass. For example, create **/event/billing/product/fee/cycle/cycle_forward_10days** to define a cycle forward event that occurs every ten days.

   > **Note**
   >
   > Use the **/event/billing/product/fee/cycle/cycle_forward_quarterly** object specification as a model.

Copyright © 2017, 2026, Oracle and/or its affiliates.

2. Map the new event to a valid purchase level:
   a. In the service-event map configuration file (*BRM_home*/**sys/data/pricing/example/pin_event_map**), add an entry for the new cycle forward event. The entry must use the following format:

      ```
      purchase_level:event_type:event_description:count: unit
      ```

      where:

      - *count* specifies the frequency of the cycle. It must be a positive number.
      - *unit* must be **day**, **week**, **month**, or **year**.

      For example, to map a biannual (24-month duration) cycle forward event to an account purchase type, the **pin_event_map** entry is:

      ```
      /account:/event/billing/product/fee/cycle/cycle_forward_biannual:Biannual Cycle Forward Event:24:month
      ```

      or

      ```
      /account:/event/billing/product/fee/cycle/cycle_forward_biannual:Biannual Cycle Forward Event:2:year
      ```

      See "Creating Service and Event Storable Classes" in *BRM Developer's Guide*.
   b. Run the **load_event_map** utility.
3. Map the new event type to a valid ratable usage metric (RUM):
   a. In the usage map configuration file, add an entry for the new cycle forward event. For example:

      ```
      /event/billing/product/fee/cycle/cycle_forward_biannual:Occurrence: 0: 0: 0: 0: 0: 0: 0: cycle_forward_biannual
      ```
   b. Run the **load_usage_map** utility.
4. Stop and restart the CM.
5. Use PDC to create charge offers that include your new cycle forward fee.

Charging Cycle Forward Fees Associated with Flexible Cycles

Cycle forward fees are charged when you run monthly billing or the **pin_cycle_forward** billing utility.

When a cycle forward event is generated, balance impacts are applied using balance validity dates.

If a cycle forward event balance impact for a noncurrency balance is set up with a relative cycle start date, balance impacts are applied either to the current cycle or to a future cycle.

For example, if the relative cycle is set to **1** and the cycle is from January 1 to March 1, a sub-balance is created with a validity period from January 1 to March 1. If the relative cycle is set to **2**, a sub-balance is created with a validity period from March 1 to May 1.

Applying Cycle Forward Fees in Parallel

For accounts with multiple services (for example, a wholesale market customer account), you can configure BRM to apply cycle forward fees in parallel for multiple services instead of applying cycle forward fees sequentially for each service, thereby reducing the time to complete billing.

Copyright © 2017, 2026, Oracle and/or its affiliates.

When you configure BRM to apply cycle forward fees in parallel, you can also configure BRM to:

- Enforce cycle fee processing prior to billing. By doing so, BRM eliminates the process of applying cycle forward fees during billing and improves the performance of the billing process.
- Use a single item at the account level to accumulate the cycle charges for all the services. By doing so, BRM reduces the number of items to process and improves overall system performance, which is important when you bill for wholesale customer accounts. When BRM applies cycle forward fees in parallel with service charges aggregated to a single account item, the account can have only a single bill unit (**/billinfo** object). Even though the account item aggregates the service charges, the respective service balance groups are still updated with the service charges.

Before configuring BRM to apply cycle forward fees in parallel, your configuration must meet these requirements:

- The number of services attached to a single balance group must be less than ten to get the performance benefit of applying cycle forward fees in parallel.
- There should be no dependency on the order of applying cycle forward fees for account groups (hierarchical, charge sharing, or discount sharing). This is because the cycle forward fees are applied by the **pin_cycle_fees** utility instead of by the billing application that gives more control over the order of processing accounts in hierarchies.

Applying parallel cycle forward fees involves the following processes:

1. Running the **pin_cycle_fees** utility in parallel at the services level, which processes cycle forward fees aligned to the accounting cycle.

   > **Note**
   >
   > When the parallel fee processing feature is enabled, cycle forward fees are applied by the **pin_cycle_fees** utility, which is run before the **pin_deferred_act** utility. However, when the feature is not enabled, cycle forward fees are applied by the billing application after running **pin_deferred_act**.

2. Running the **pin_update_journals** utility to post-process cycle forward fees.
3. Running the **pin_bill_accts** utility for regular billing.
4. Running the **pin_cycle_fees** utility to process flexible cycle forward fees.
5. Running the **pin_update_journals** utility to post-process flexible cycle forward fees.

To configure BRM to apply cycle forward fees in parallel, run the **pin_bus_params** utility to change the **StagedBillingFeeProcessing** business parameter. For information about this utility, see "pin_bus_params" in *BRM Developer's Guide*.

1. Go to *BRM_home*/**sys/data/config**.
2. Create an XML file from the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsBilling bus_params_billing.xml
   ```

3. Search for this line:

   ```
   <StagedBillingFeeProcessing>0</StagedBillingFeeProcessing>
   ```

   The default is **0**. (BRM applies the cycle forward fees as part of the billing process.)

Copyright © 2017, 2026, Oracle and/or its affiliates.

> **Note**
>
> The cycle fee processing at the time of billing is only with reference to the cycle forward fees aligned with the accounting cycle boundary.

4. Do one of these:
   - To apply cycle forward fees in parallel by service, specify **1**.
   - To enforce cycle fee processing prior to billing and apply cycle forward fees in parallel by service, specify **2**.

     > **Note**
     >
     > If you set **StagedBillingFeeProcessing** to **2**, you will not be able to use billing time discounts, cycle folds, or cycle rollovers.

   - To apply cycle forward fees in parallel by service with service charges aggregated to a single account item, specify **3**.
5. Save the file as **bus_params_billing.xml**.
6. Load the XML file into the BRM database:

   ```
   pin_bus_params bus_params_billing.xml
   ```

7. Stop and restart the CM.

About Enforcing Cycle Forward Fee Processing Prior to Billing

When BRM enforces cycle fee processing prior to billing, the following processes are impacted:

- The **pin_cycle_fees** utility performs additional error processing to set the error status (as needed) on the corresponding bill unit

  > **Note**
  >
  > If you customize **pin_cycle_fees** and use the application global structure PIN_FLD_EXTENDED_INFO provided by the multithreaded application framework to hold custom information at run time, you must consider that **pin_cycle_fees** stores the error processing information in a single array element PIN_FLD_ERROR_INFO under PIN_FLD_EXTENDED_INFO.

- The billing process stops if any of the following conditions is true:
  - The BILLING_STATUS_FLAGS field of the **/billinfo** object indicates that there was an error processing one of the cycle forward fees.
  - There is at least one service for which cycle fee processing (regular, deferred, deferred purchase, or deferred cancellation) has not been completed for the accounting cycle being billed.
- In rare cases, if billing is due for a bill unit for more than one accounting cycles, special handling is required. See "Handling Skipped Billing" for more information.

Copyright © 2017, 2026, Oracle and/or its affiliates.

About Aggregating Service Charges to Account Items

When applying cycle forward fees in parallel by service with service charges aggregated to a single account item, multiple threads of **pin_cycle_fees** updates a single item. To avoid updating the same item by multiple threads, **pin_cycle_fees** logs the item and journal updates to temporary tables as follows:

- Logs item updates to the **/tmp_events_to_process** object in the TMP_EVENTS_TO_PROCESS_T table.
- Logs journal updates to the **/tmp_journals_to_process** object in the TMP_JOURNALS_TO_PROCESS_T table.

The **pin_update_journals** utility processes the temporary item and journal data and updates the main item and journal tables.

To ensure efficient access of these temporary tables, Oracle recommends the following:

- **Resetting high-water mark.** Records are frequently inserted into and deleted from the temporary tables, which can fragment the temporary tables. Reset the high-water mark for the temporary tables as the BRM schema user.

  Run these commands every time before calling the **pin_bill_accts** inside the **pin_bill_day** script:

  ```
  ALTER TABLE TMP_JOURNALS_TO_PROCESS_T ENABLE ROW MOVEMENT;
  ALTER TABLE TMP_JOURNALS_TO_PROCESS_T SHRINK SPACE;
  ALTER TABLE tmp_events_to_process_t ENABLE ROW MOVEMENT;
  ALTER TABLE tmp_events_to_process_t SHRINK SPACE;
  ```

  For more information about the high-water mark, see the Oracle Database documentation.

- **Presetting statistics.** Preset the statistics of the temporary tables that are created during BRM installation by running the following commands as a one-time activity. This enables BRM to avoid a full scan of these tables.

  ```
  Exec
  dbms_stats.set_table_stats('SCHEMA_NAME','TMP_EVENTS_TO_PROCESS_T','','','',200000000
  ,40000000,1250) ;

  Exec
  dbms_stats.set_index_stats('SCHEMA_NAME','I_TMP_EVENTS_ID','',numrows=>200000000,numl
  blks=>1000000,numdist=>200000000,avglblk=>1,avgdblk=>1,clstfct=>200000000);

  Exec
  dbms_stats.set_column_stats('SCHEMA_NAME','TMP_EVENTS_TO_PROCESS_T','POID_ID0','',dis
  tcnt=>200000000,density=>1/200000000,nullcnt=>0,srec=>srec_eve,avgclen=>11);

  Exec
  dbms_stats.set_table_stats('SCHEMA_NAME','TMP_JOURNALS_TO_PROCESS_T','','','',2000000
  00,40000000,1250);

  Exec
  dbms_stats.set_index_stats('SCHEMA_NAME','I_TMP_JOURNALS_ID','',numrows=>200000000,nu
  mlblks=>1000000,numdist=>200000000,avglblk=>1,avgdblk=>1,clstfct=>200000000);

  Exec
  dbms_stats.set_column_stats('SCHEMA_NAME','TMP_JOURNALS_TO_PROCESS_T','POID_ID0','',d
  istcnt=>200000000,density=>1/200000000,nullcnt=>0,srec=>srec_eve,avgclen=>11);

  Exec dbms_stats.lock_table_stats('SCHEMA_NAME','TMP_EVENTS_TO_PROCESS_T');
  ```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
Exec dbms_stats.lock_table_stats('SCHEMA_NAME','TMP_JOURNALS_TO_PROCESS_T');
```

where *SCHEMA_NAME* is the BRM database user, such as **pin1**.

Handling Skipped Billing

In rare cases, if billing is due for a bill unit for more than one accounting cycles, special handling is required. This multiple cycle overdue billing is referred to as skipped billing.

For example, assume that the current date is December 1 and BRM did not perform billing for the cycles ending November 1 and October 1. In this case, when you run **pin_bill_day** on the current date, three bills are due to be created.

When BRM tries to calculate the cycle forward fees, the following happens:

- **pin_cycle_fees** applies cycle forward fees due only as of October 1 because October 1 billing has not been processed yet.
- **pin_bill_accts** performs billing only on October 1 and stops with an error when performing billing on November 1 because cycle forward fees due as of November 1 have not been processed yet.

To handle the case of skipped billing used in this example:

1. Run **pin_cycle_fees**, **pin_update_journals**, and **pin_deferred_act** in the following sequence:

   ```
   pin_cycle_fees -defer_purchase
   pin_cycle_fees -defer_cycle_fees
   pin_cycle_fees -defer_cancel
   pin_cycle_fees -regular_cycle_fees
   pin_update_journals
   pin_deferred_act
   ```

2. Run **pin_bill_accts**.
3. Repeat step 1 and step 2 twice more, which performs billing for November 1 and December 1.

Using the pin_bill_day Script to Apply Parallel Cycle Forward Fees

To support applying cycle forward fees in parallel, the **pin_bill_day** script includes the following commented out sections:

- **Pre-Billing Parallel Cycle Fee Processing**: Includes the following entries for **pin_cycle_fees** and **pin_update_journals**:

  ```
  ##### pin_cycle_fees -defer_purchase
  ##### pin_cycle_fees -defer_cycle_fees
  ##### pin_cycle_fees -defer_cancel
  ###### pin_cycle_fees -regular_cycle_fees
  ##### pin_update_journals
  ```

- **Post-Billing Parallel Cycle Fee Processing**: Includes the following entry for **pin_update_journals**:

  ```
  ##### pin_update_journals
  ```

To apply cycle forward fees in parallel by using the **pin_bill_day** script:

1. Make sure that the **StagedBillingFeeProcessing** parameter is not set to **0**.

Copyright © 2017, 2026, Oracle and/or its affiliates.

2. Open the *BRM_home*/**bin/pin_bill_day** script in a text editor.
3. Uncommment these entries in the **Pre-Billing Parallel Cycle Fee Processing** section:

```
##### pin_cycle_fees -defer_purchase
##### pin_cycle_fees -defer_cycle_fees
##### pin_cycle_fees -defer_cancel
###### pin_cycle_fees -regular_cycle_fees
##### pin_update_journals
```

4. Uncomment this entry in the **Post-Billing Parallel Cycle Fee Processing** section:

```
##### pin_update_journals
```

5. Save and close the file.
6. Run **pin_bill_day**.

About Limitations and Impacts of Applying Cycle Forward Fees in Parallel

Configuring BRM to apply cycle forward fees in parallel includes these limitations and impacts:

- In rare cases, when the **pin_cycle_fees** utility successfully creates temporary item and journal data and the subsequent run of the **pin_update_journals** utility fails to update the item and journal tables, you must investigate and correct the problem in processing the temporary item and journal data before performing any accounts receivable action or generating ledger reports.
- If the parallel fee processing feature is configured to enforce cycle fee processing before billing, balance impact events that occur before the **pin_cycle_fees** utility runs are cancelled with an error.
- If the parallel fee processing feature is *not* configured to enforce cycle fee processing before billing, balance impact events that occur before the **pin_cycle_fees** utility runs result in triggered billing that can be slow due to serial application of cycle forward fees.
- Performance of the following operations is not improved because parallel cycle fee processing does not apply to these operations:
  - Trial billing
  - Bill Now
  - Billing on purchase
  - Account creation
  - Purchase of bundles
  - Billing time discount
  - Cycle fold
  - Rollover
  - Account activation, account inactivation, or account cancellation
  - Rerating

Applying Recurring Charges Based on Charge Offer Priority

When multiple charge offers in a bundle include recurring charges, you can configure BRM to apply recurring charges in the order of charge offer priority. You can apply recurring charges based on charge offer priority during the following operations:

Copyright © 2017, 2026, Oracle and/or its affiliates.

- Bundle purchase or cancellation, for all the charge offers per bundle.
- Billing, for all the charge offers in a bundle per bill unit.

> **Important**
>
> This parameter does not prioritize charge offers for recurring charges applied by **pin_cycle_fees -defer_cancel** and does not prioritize customized charge offers.

To enable this feature, run the **pin_bus_params** utility to change the **UsePrioritySubscriptionFees** business parameter. For information about this utility, see "pin_bus_params" in *BRM Developer's Guide*.

To apply recurring charges based on charge offer priority:

1. Go to *BRM_home*/**sys/data/config**.
2. Create an XML file from the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsSubscription bus_params_subscription.xml
   ```

3. In the file, change **disabled** to **enabled**:

   ```
   <UsePrioritySubscriptionFees>enabled</UsePrioritySubscriptionFees>
   ```

4. Save the file as **bus_params_subscription.xml**.
5. Load the XML file into the BRM database:

   ```
   pin_bus_params bus_params_subscription.xml
   ```

6. Stop and restart the CM.

Customizing the Cycle Interval for Specified Charge Offers

To customize the time interval for applying cycle forward and cycle arrears fees for a specified charge offer, 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 charge offer. The start and end dates provided are used to calculate the scale to determine the cycle fee amount to charge or refund.

For example, if a charge offer is purchased on April 1 with a monthly cycle forward fee of $30 and the purchase, usage, and cycle end dates are set to April 20, the cycle fee amount is based on the scale for the period April 1 to April 20 (20 days) divided by the unit interval from April 1 to April 30 (30 days). The cycle fee charged is **20/30** * $30, or $20.

If the charge offer is canceled on April 15, the refund amount is based on the scale for the period April 15 to April 20 (5 days) divided by the unit interval from April 1 to April 20 (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 charge offer was valid during the cycle.

Copyright © 2017, 2026, Oracle and/or its affiliates.

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.

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.

Copyright © 2017, 2026, Oracle and/or its affiliates.

22

Working with Extended Rating Attributes

Learn how to apply special charges or discounts based on a specific attribute of a service or account by creating extended rating attributes (ERAs) in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

- About Extended Rating Attributes
- Creating Friends and Family ERAs
- Default ERAs

About Extended Rating Attributes

Extended rating attributes (ERAs) provide special charges 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.

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 ERAs only.

For more information, see "Creating and Managing Profile Sharing Groups" in *BRM Managing Customers*.

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.

Default ERAs

Table 22-1 summarizes the default account-level ERAs.

Table 22-1 Default ERAs

Function	Name
Assign an account to a business segment for business intelligence reporting.	Business Intelligence Segment ERA
Assign an account to a customer group, or assign an arbitrary quality level to an account.	Customer type or quality ERA

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 22-1 (Cont.) Default ERAs

Function	Name
Assign an account to a group of accounts that share certain telephony properties.	Corporate agreement ERA
Create a closed user group, such as a group of all mobile numbers in a company.	Closed user group ERA
Give a discount for calls made on a specific calendar date.	Special day discount ERA
Define the home cells for a customer and allow discounts while calling from this area.	Home cell assignment ERA
Provide discounts to calls made to specific numbers or regions, such as all numbers in a country or area code.	Home region code ERA
Provide discounts to calls made to specific numbers, such as those for friends and family members.	Friends and family ERA
Assign a Quality of Service to a service.	Service-level agreement ERA

Copyright © 2017, 2026, Oracle and/or its affiliates.

23

Configuring Closed User Groups

Learn how to configure closed user groups for Oracle Communications Billing and Revenue Management (BRM) in Pricing Design Center (PDC).

Topics in this document:

- Configuring Closed User Groups

Configuring Closed User Groups

You can configure closed user groups for applying special prices for events that originate and terminate between a list of subscribers. For example, you can configure a closed user group for applying special rates on calls between employees of a small company.

To identify the members of a closed user group and apply special prices, you configure a custom rule and add it to a generic selector. The custom rule evaluates the groups to which the originating and terminating subscribers belong. If the originating and terminating subscribers belong to even one common closed user group, the custom rule evaluates as true. The generic selector guides the price based on how the custom rule is evaluated at runtime.

You can configure closed user groups at the account level or at the service level. To configure closed user groups at the account level, use the **OOB_ProfileSpecifications.xml** and **OOB_CRs.xml** files as templates. To configure closed users groups at the service level, use the **Sample_ServiceCUG_ProfileSpecification.xml** and **Sample_ServiceCUG_CR.xml** files as templates. See Table 23-1 for the usage and description of each element.

If you configure a closed user group at the account level, associate the closed user group names with the appropriate service identifiers for the account when the customer purchases the charge offer with a closed user group. For example, map the phone numbers to the appropriate closed user group names.

If you configure a closed user group at the service level, configure only the appropriate closed user group names when the customer purchases the charge offer with a closed user group.

Table 23-1 lists the elements in the custom rules XML file, the usage of each element, and a description of how to specify each element in the XML file.

Table 23-1 Elements in the Custom Rules XML File

Element	Syntax	Description
customAnalyzerRules	`<customAnalyzerRules>`	The root element of the rule.
name	`<name>`*`RuleName`*`</name>`	*RuleName* specifies the custom rule name.
description	`<description>`*`RuleDescription`*`</description>`	*RuleDescription* contains the description of the custom rule.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 23-1 (Cont.) Elements in the Custom Rules XML File

Element	Syntax	Description
priceListName	`<priceListName>PriceListName </priceListName>`	*PriceListName* specifies the price list name. BRM uses only one internal price list. The only value is **Default**.
profileSpecName	`<profileSpecName>ProfileAttributeSpecificationName</profileSpecName>`	*ProfileAttributeSpecificationName* specifies the name of the profile attribute specification used for configuring the rule.
customRuleExpression	`<customRuleExpression>` `  <dynamicFieldCRExpression>` `  <operator>Operator</operator>` `  <FieldIdentifier>` `...` `  </FieldIdentifier>` `  </dynamicFieldCRExpression>` `</customRuleExpression>`	Contains the expression to evaluate closed user groups, where *Operator* specifies the operator to use in the expression. The valid value is **INTERSECT**.
FieldIdentifier	`<FieldIdentifier>` `  <qualifier>Qualifier</qualifier` `  <fieldSelector>` `...` `  </fieldSelector>` `</FieldIdentifier>`	Contains the details to identify the fields for evaluating closed user groups, where *Qualifier* specifies the qualifier for identifying the fields. The valid values are: • **ORGINATING** • **TERMINATING** In the **<FieldIdentifier>** element, you can define fields from the service, account, and event attribute specifications.
FieldSelector	`<fieldSelector>` `  <operator>ConditionOperator</operator>` `  <selectorFieldName>SelectorFieldName</selectorFieldName>` `  <selectorFieldKind>SelectorFieldKind</selectorFieldKind>` `  <conditionFieldName>ConditionFieldName</conditionFieldName>` `  <conditionFieldKind>ConditionFieldKind</conditionFieldKind>` `</fieldSelector>`	Identifies the closed user group names that match the given product identifier (PUID); for example, a phone number, for both the originating and terminating subscribers. **Note:** You can use the **<FieldSelector>** element for configuring account-level closed user groups only. where: • *ConditionOperator* specifies the operator to use in the condition. The valid value is **EQUAL_TO**. You must specify the *ConditionOperator* if you want to define the conditions for selecting fields. If the *ConditionOperator* is null, you cannot specify *ConditionFieldName* and *ConditionFieldKind*. • *SelectorFieldName* specifies the selector field name. You must define the selector fields based on the condition fields. • *SelectorFieldKind* specifies the type of attributes used for field selection. You can specify any service, event, or account attribute. • *ConditionFieldName* specifies the condition field name. • *ConditionFieldKind* specifies the type of attributes used for selection. You can specify any service, event, or account attribute.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Part IV
Managing Balances

This part describes how to manage balances using Oracle Communications Billing and Revenue Management (BRM) and Pricing Design Center (PDC).

This part contains the following chapters:

- Managing Balances During Charging
- Managing Sub-Balances
- Configuring Dynamic Quota
- About Folds
- Applying Credit Limits to Balances
- About Rollovers
- Configuring Balance Impact Rounding

Copyright © 2017, 2026, Oracle and/or its affiliates.

24

Managing Balances During Charging

Learn how balances are managed in Oracle Communications Billing and Revenue Management (BRM) product offerings created using Pricing Design Center (PDC).

Topics in this document:

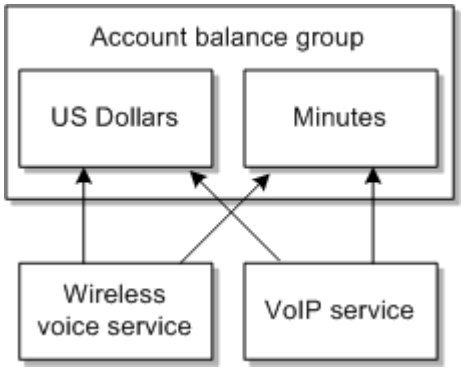
- Tracking Balances in Balance Groups

Tracking Balances in Balance Groups

Using a service can affect multiple types of balances. For example, a phone service might be charged in both dollars and minutes. In this case, the service usage would be associated with a balance group that includes two balances, one for dollars and one for minutes. *Balance groups* are collections of multiple balances that apply to one or more services.

You create balance groups when you create packages in PDC. By default, a package contains one balance group: the account balance group. Balances for every service in the package share that balance group, which means that balances such as included minutes are shared among all services. Figure 24-1 shows a default balance group with two types of balances, both of which are used by two services.

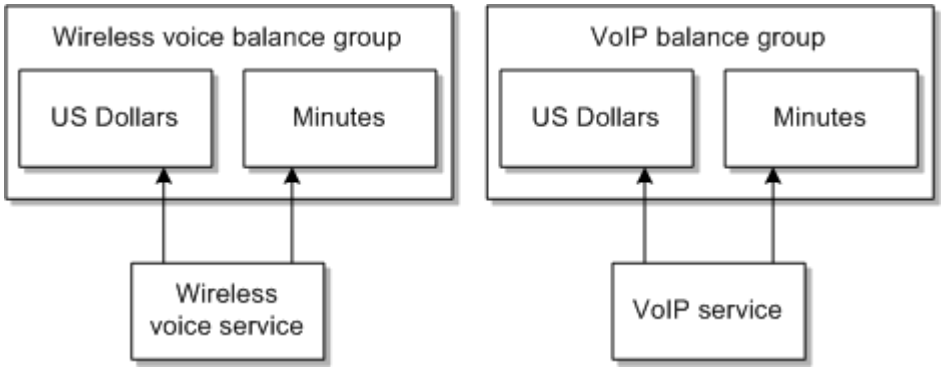
Figure 24-1 Default Balance Group for Two Services



To track and control the allocation and consumption of balances for specific services, you can create multiple balance groups and assign services to their own balance groups, or group sets of services by balance group. Figure 24-2 shows two services that each use their own balance group.

Copyright © 2017, 2026, Oracle and/or its affiliates.

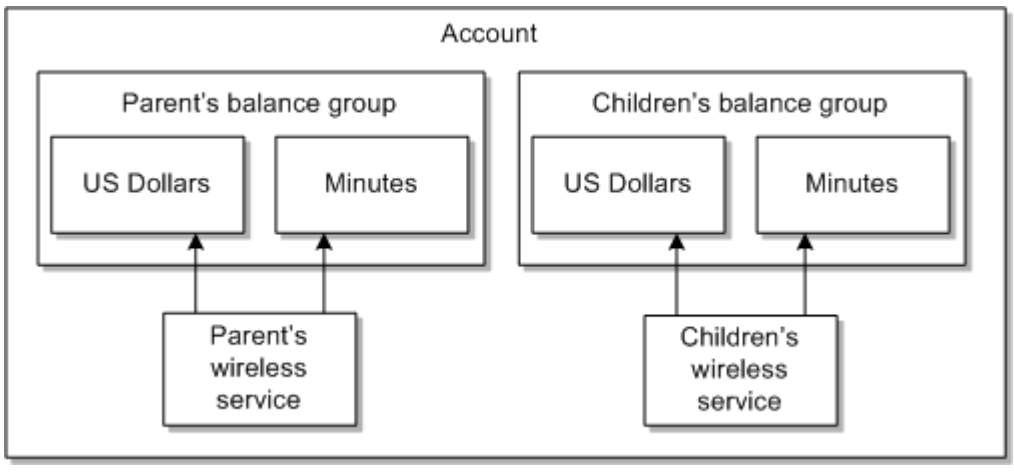
Figure 24-2 Two Balance Groups for Two Services



In addition to creating different balance groups for different services, you can create different balance groups for different packages that are purchased by the same account. For example, consider a family of four that has a mobile phone service for each family member. The bundle for each service includes 300 minutes. If the package has only the default account balance group, all four family members' included minutes go into the same account balance, which would contain 1200 included minutes. One teenager could then use 1000 minutes, leaving only 200 minutes to be shared by the other three family members.

To avoid that situation, you could create a package with two mobile phone services in the same bundle for the parents and an optional add-on package in its own balance group. When the parents purchase add-on packages for the children, each child's included minutes are tracked separately in a separate balance group. Figure 24-3 shows two balance groups, created by purchasing separate packages in the same account.

Figure 24-3 Separate Balance Groups By Purchasing Separate Packages



You can set different credit limits for the same balance element in each balance group. For example, the parents might request a credit limit of $10 on the US Dollar balance element in each child's balance group to control overage charges but request a $100 credit limit on the US Dollar balance in the balance group for their mobile phone services. See "Applying Credit Limits to Balances" for more information.

Copyright © 2017, 2026, Oracle and/or its affiliates.

25

Managing Sub-Balances

Learn how to manage sub-balances in Oracle Communications Billing and Revenue Management (BRM) product offerings.

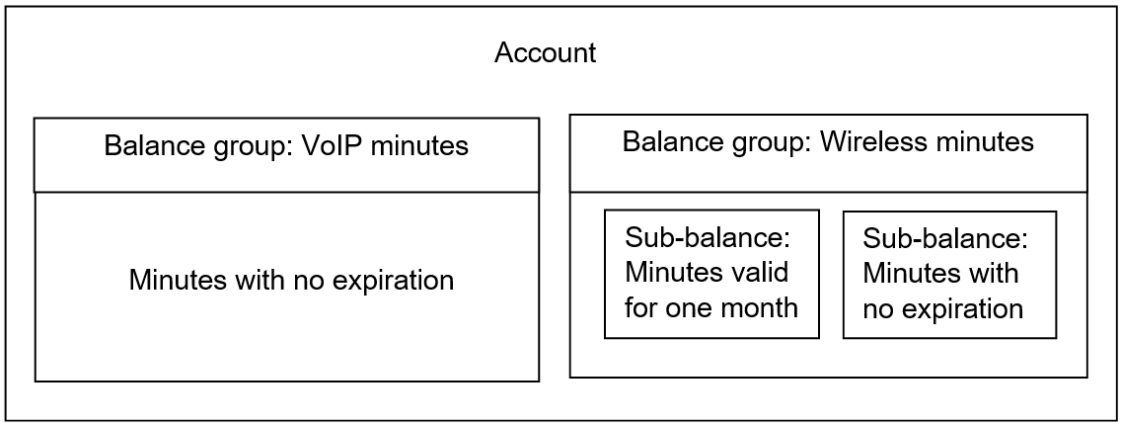
Topics in this document:

- About Sub-Balances
- About Noncurrency Sub-Balances
- How Balances in Validity-Based Sub-Balances Are Updated
- Restricting the Validity of Noncurrency Sub-Balances That Start on First Usage
- Configuring Custom Sub-Balances
- Specifying the Order in Which Sub-Balances Are Consumed
- Configuring Time-Stamp Rounding for Cycle Events
- Configuring Time-Stamp Rounding for Purchase Events

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 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. Figure 25-1 shows an account with two balance groups, one for each service. In the balance group for wireless minutes, there are balances with different expiration dates. Therefore, there are two sub-balances for wireless minutes.

Figure 25-1 Balance Group With Sub-Balances



A currency sub-balance can store the balances for multiple services. For example, an account that owns two charge offers that cost $25.00 per offer has a starting currency sub-balance

Copyright © 2017, 2026, Oracle and/or its affiliates.

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 subscription renewal 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.
  Balances with the same validity periods are stored in the same sub-balance, while balances with unique validity periods are stored in separate sub-balances. For more information, see "How Balances in Validity-Based Sub-Balances Are Updated".
- The current amount of the sub-balance.
- 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 minutes for a phone call, the session object is specified. A separate sub-balance is kept for each unique contributor.
- Rollover data, such as the rollover period and the balance amount that is rolled over, if any. See "About Rollovers".
- The ID of the offer that granted the balance (referred to as the “grantor object.")

If the **TimestampRounding** business parameter is enabled, the start time of the balances granted by cycle events is rounded to midnight. See "Configuring Time-Stamp Rounding for Cycle Events". However, the start time of the balances granted by purchase events is not rounded to midnight unless the **TimeStampRoundingForPurchaseGrant** business parameter is also enabled. See "Configuring Time-Stamp Rounding for Purchase Events".

You can configure sub-balances to track various types of balances and usage. See "Configuring Custom Sub-Balances".

About Noncurrency Sub-Balances

A noncurrency sub-balance typically has a limited validity period (for example, the period during which minutes can be used). Noncurrency sub-balances can contain various types of balances, such as the following:

- Minutes
- Frequent flyer miles
- Loyalty points
- Number of emails or text messages

A noncurrency sub-balance can also keep track of the total balances 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 balance.

When granting a noncurrency balance, 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

Copyright © 2017, 2026, Oracle and/or its affiliates.

- 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 "Configuring Custom Sub-Balances".

How Balances in Validity-Based Sub-Balances Are Updated

By default, BRM stores balances with the same validity periods in the same sub-balance, provided they are associated with the same balance group. BRM automatically creates a new sub-balance for balances with a unique validity period if one does not already exist.

For example, an account owns two services that each include 100 minutes that are always valid. The account has a balance of 200 minutes stored in a single sub-balance. When the customer uses minutes from each service, the minutes are consumed from the common sub-balance.

Balances with different validity periods are tracked in separate sub-balances. For example, an account owns two services that each include 100 minutes. Minutes for service 1 expire at the end of the month, and minutes for service 2 expire at the end of the year. Each set of minutes is stored in a separate sub-balance.

> **Note**
>
> - When noncurrency balances are configured to start on first usage, BRM creates a new sub-balance for each balance whether they have the same validity period. See "Restricting the Validity of Noncurrency Sub-Balances That Start on First Usage".
> - In BRM, during rating, for noncurrency resources configured with a validity date of "first usage", the system creates two sub-balances for each resource: one for the scaled amount and one for the fixed amount.

You can specify the order in which sub-balances are consumed by setting up balance consumption rules. See "Specifying the Order in Which Sub-Balances Are Consumed".

You can also limit how validity-based balances, such as minutes, are summed by configuring sub-balances. For example, you might want to limit usage of minutes to a specific service or a specific call session. See "Configuring Custom Sub-Balances".

You can configure BRM to round time stamps to midnight for balances granted by cycle and purchase events. See the following:

- Configuring Time-Stamp Rounding for Cycle Events
- Configuring Time-Stamp Rounding for Purchase Events

For more information, see "Configuring Timestamp Rounding" in *BRM Configuring and Running Billing*.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Restricting the Validity of Noncurrency Sub-Balances That Start on First Usage

When a noncurrency balance is configured to start on first usage (when the customer first uses the service), BRM always creates a new sub-balance for that balance when it is granted. A new sub-balance is created for each balance even when a charge offer or discount offer grants multiple first-usage balances of the same type. A balance that has a first-usage start time will remain available for consumption for as long as the balance is not used.

> **Note**
>
> If a balance is shared among accounts in a sharing group, the balance validity period is set when any account in the group first impacts the balance. Because the same balance is shared with all accounts in the group, the validity period of that balance applies to all accounts.

You can configure BRM to automatically restrict the validity period of granted noncurrency balances to end no later than the end time of the charge offer or discount offer that grants the balance. Restricting the balance end time to the offer's end time ensures that the balance cannot continue to be consumed after the offer expires.

> **Note**
>
> When an offer is *canceled*, the validity period end time of balances granted by that charge offer or discount offer is set to the time of the cancellation.

To enable this feature, run the **pin_bus_params** utility to change the **RestrictResourceValidityToOffer** business parameter. See "pin_bus_params" in *BRM Developer's Guide* for more information.

To restrict balance validity end times to charge offer or discount offer end times:

1. Go to *BRM_home*/**sys/data/config**.
2. Create an XML file from the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsMultiBal bus_params_multi_bal.xml
   ```

3. In the file, change **FALSE** to **TRUE**:

   ```
   <RestrictResourceValidityToOffer>TRUE</RestrictResourceValidityToOffer>
   ```

4. Save the file as **bus_params_multi_bal.xml**.
5. Load the XML file into the BRM database:

   ```
   pin_bus_params bus_params_multi_bal.xml
   ```

6. Stop and restart the CM.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Configuring Custom Sub-Balances

As described in "About Sub-Balances", BRM automatically creates sub-balances when portions of the balance are valid at different times. You do not need to configure sub-balances based on validity times.

You can configure custom sub-balances to track balances based on other values. For example:

- Minutes per call session
- Frequent flier miles per service instance
- Friends and family calls to specific locations

You configure custom sub-balances by editing and loading a configuration file.

To customize the allocation and consumption of balances, you configure sub-balances by specifying the following values:

- **Balance element ID**. This is the ID for the type of balance in the sub-balance, such as dollars or 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 from an object related to the charge, such as a service object or account object.

For example, you might configure a custom sub-balance to track balance impacts for:

- Dollars
- Charged by GPRS events
- Tracked by each GPRS session

Sub-balance contributors define how to track sub-balances. For example, if the contributor is a service object, sub-balances are tracked separately for each service instance.

Sub-balance contributors are specified by a field name from the event, or from an object related to the charge, such as a service object or account object. Some fields you might want to use as contributors include:

- The service object

  Specify a service field to track sub-balances 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 balances. The other way is to create a balance group for the service when you create your packages in PDC.

- The session object

  Specify a session field to track balances per session instance. This is useful when you offer discounts for certain levels of usage (for example, 10 frequent flier miles per hour of phone calls).

- The charge offer object

Copyright © 2017, 2026, Oracle and/or its affiliates.

When several charge offers in the same package have different rollover rules, tracking balances per charge offer permits BRM to update the minutes for each charge offer.

- The phone number field

  Specify a phone number field to track balances for called telephone numbers.

- The account object

  Specify the account field to track balances that are shared among several accounts, such as earned minutes. When minutes are distributed, they can be divided based on each account's usage level.

There are two types of contributors:

- **Retrieving contributor**. This is the field from which BRM retrieves the amount to add to the sub-balance. For example, if the retrieving contributor is the service for a GPRS event, a separate balance summary is retrieved from each unique GPRS service object.

  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**. Whereas the retrieving contributor determines the balance amount to retrieve from a charge, the updating contributor specifies how to apply the amount to a sub-balance. For example, to add or deduct minutes for specific broadband sessions, specify the session object as the updating contributor.

Table 25-1 shows contributor configurations to track balances for GSM services; in this case, one sub-balance for dollars and one sub-balance for minutes.

Table 25-1 Contributor Configurations

Balance Element	Event Type	Retrieving Contributor	Updating Contributor
Dollars	**/event/session/gsm**	PIN_FLD_SERVICE_OBJ	PIN_FLD_SERVICE_OBJ
Minutes	**/event/session/gsm**	PIN_FLD_SERVICE_OBJ	PIN_FLD_SERVICE_OBJ

Each row in this example specifies a sub-balance:

- Dollars balance impact
  - **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 sub-balance.
  - PIN_FLD_SERVICE_OBJ is the retrieving contributor, so the dollars balance 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.
- Minutes balance impact
  - **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 minutes balance.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- PIN_FLD_SERVICE_OBJ is the retrieving contributor, so the minutes balance 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, 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 minutes for every unique GSM service that includes minutes. Additional minutes and consumption of minutes are restricted to the GSM service instance.

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 balance consumption rules. See "Specifying the Order in Which 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 Custom Sub-Balances".

Defining and Loading Custom Sub-Balances

See "Configuring Custom Sub-Balances" for information.

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.

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:

```
balance_type:event_type:retrieving_contributor:updating_contributor
```

For example:

```
1000003:/event/session/gprs:PIN_FLD_SESSION_OBJ:PIN_FLD_SESSION_OBJ
```

where:

- **1000003** is the balance element ID for frequent flyer miles.
- **/event/session/gprs** is the event type that impacts the frequent flyer miles balance.
- **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 (balance element type 840) in a single sub-balance for all events with any contributor:

```
840:/event:*:*
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

To configure several sub-balances for one balance element, leave subsequent balance element fields empty. If no balance element is specified, the previous balance element specified is used. For example, the next two configurations use the same balance element (100002):

```
100002 : /event/session/telco/gsm : PIN_FLD_SESSION_OBJ : PIN_FLD_SESSION_OBJ
       : /event/session/call : PIN_FLD_SERVICE_OBJ : PIN_FLD_SERVICE_OBJ
```

After editing the file, you load it into the BRM database.

> **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.

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_contributors".

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.

Specifying the Order in Which Sub-Balances Are Consumed

In a balance group, balances for the same type of balance element are combined into a single balance. For example, an account that owns two charge offers that each include 300 minutes has a starting included minute balance of 600 minutes, providing the services are associated with the same balance group and have the same validity period for the included minutes.

When portions of a balance are valid at different times, however, BRM creates multiple sub-balances for that balance in the balance group. For example, a balance group might include 300 minutes that are valid only for the current billing cycle and 1000 minutes that never expire. Because the included minutes have different validity periods, they are tracked in different sub-balances.

When a customer uses a service, BRM must know which sub-balance to use first. To specify the order in which sub-balances are consumed, you use consumption rules. The rules are based on the start and end times of the sub-balances, for example consume the sub-balance with the earliest validity start time first.

For example, you can specify whether a subscriber with the following minute sub-balances uses the Anytime Minutes or the rollover Anytime Minutes first:

- 100 Anytime Minutes that are valid March 1 to April 30
- 50 rollover Anytime Minutes that are valid February 1 to March 30

Copyright © 2017, 2026, Oracle and/or its affiliates.

If the minute balance is associated with a rule that says to consume the sub-balance that expires first, the 50 rollover Anytime Minutes are used first.

If the minute balance is associated with a rule that says to consume the sub-balance with the latest start time, the 100 Anytime Minutes are used first.

Table 25-2 lists the balance consumption rules that BRM supports.

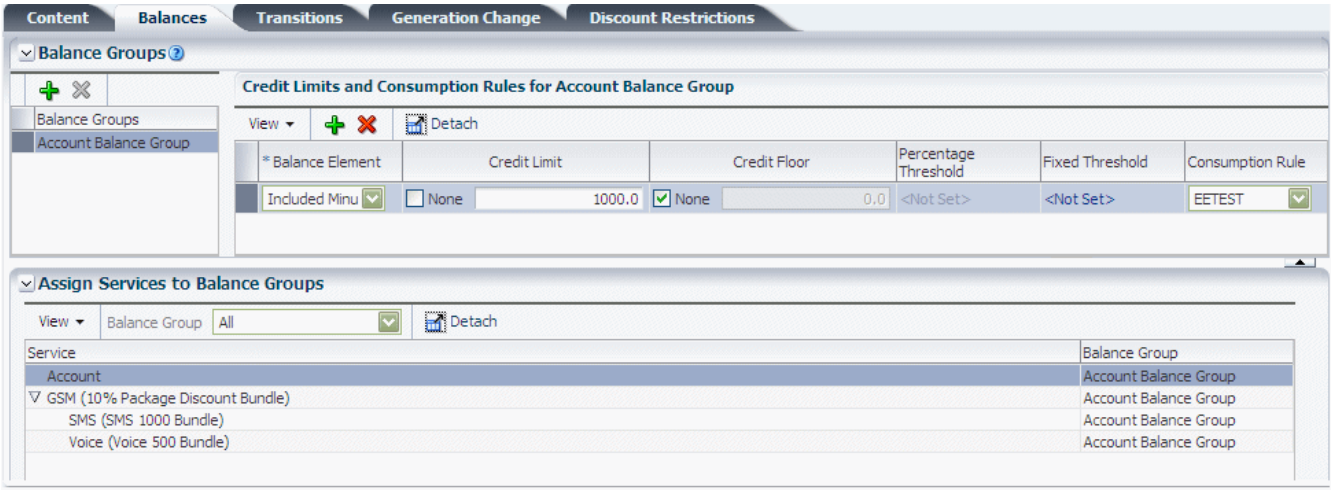
Table 25-2 Supported Consumption Rules

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.
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.

As shown in Figure 25-2, you associate consumption rules with balances in packages.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Figure 25-2 Consumption Rules for Balance Groups



You can also set systemwide and default balance element 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. **Package consumption rules**. Your product offerings can include consumption rules for each service that you support, defined differently in each package. This enables you to have different consumption rules for the same balance based on the packages purchased by the customer. When a customer purchases a package, the package's balance element-to-consumption rule mapping is stored in the customer's **/balance_group** object. If there are any conflicting rules for the same balance element, BRM uses the rule from the most recently purchased package.
3. **Systemwide service consumption rules**. You can specify a systemwide consumption rule for each service that you support. BRM uses the systemwide settings only if a rule is not defined for the balance element in the customer's purchased packages. BRM stores systemwide settings in the **/config/beid** object.
   You define systemwide consumption rules for each balance element that you support by using the **pin_beid** file and **load_pin_beid** utility. See "load_pin_beid" in *BRM Setting Up Pipeline Pricing* for more information.
4. **Default consumption rule**. You can specify a default consumption rule that applies to all balance elements. This setting is used only if a consumption rule is not defined for the balance element in the customer's purchased packages or if there is not a systemwide service rule. BRM stores the default balance element setting in the **/config/business_params** object.
   You define the default consumption rule in the **/config/business_params** object. See "Setting the Default Consumption Rule" for more information.

Setting the Default Consumption Rule

The default consumption rule applies to all balance elements in your system. The default setting is earliest start time and earliest expiration time (ESTEET). You can change this setting by running the **pin_bus_params** utility to change the **SortValidityBy** business parameter. See "pin_bus_params" in *BRM Developer's Guide* for more information.

Copyright © 2017, 2026, Oracle and/or its affiliates.

To set the default balance element consumption rule:

1. Go to *BRM_home*/**sys/data/config**.
2. Create an XML file from the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsMultiBal bus_params_multi_bal.xml
   ```

3. In the file, change **ESTEET** to one of the values listed in Table 25-2:

   ```
   <SortValidityBy>ESTEET</SortValidityBy>
   ```

4. Save the file as **bus_params_multi_bal.xml**.
5. Load the XML file into the BRM database:

   ```
   pin_bus_params bus_params_multi_bal.xml
   ```

You do not need to restart the CM to update this entry.

Configuring the Default Consumption Rule in ECE

When ECE receives a usage request for which no consumption rules are configured, ECE applies its own systemwide default consumption rule for processing the usage request.

To configure the default consumption rule in ECE:

1. Access the ECE configuration MBeans:
   a. Log on to the driver machine.
   b. Start the ECE charging servers (if they are not started).
   c. Connect to the ECE charging server node enabled for JMX management.

      This is the charging server node set to **start CohMgt = true** in the *ECE_home*/**oceceserver/config/eceTopology.conf** file.
   d. Start a JMX editor that enables you to edit MBean attributes, such as JConsole.
   e. In the editor's MBean hierarchy, find the ECE configuration MBeans.
2. Expand the **ECE Configuration** node.
3. Expand **charging.server**.
4. Expand **Attributes**.
5. Set the **systemConsumptionRule** attribute to one of the following systemwide consumption rules:
   - **EARLIEST_START**
   - **LATEST_START**
   - **EARLIEST_EXPIRATION**
   - **LATEST_EXPIRATION**
   - **EARLIEST_START_LATEST_EXPIRATION**
   - **EARLIEST_START_EARLIEST_EXPIRATION**
   - **LATEST_START_LATEST_EXPIRATION**
   - **LATEST_START_EARLIEST_EXPIRATION**
   - **EARLIEST_EXPIRATION_EARLIEST_START**
   - **EARLIEST_EXPIRATION_LATEST_START**

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **LATEST_EXPIRATION_EARLIEST_START**
- **LATEST_EXPIRATION_LATEST_START**
- **NONE**: When the attribute is set to **NONE**, the default consumption rule is not configured, and the order for consuming balances is undefined.

By default, this attribute is set to **EARLIEST_START_EARLIEST_EXPIRATION**.

6. Save your changes.

Deleting Expired Sub-Balances

To delete expired sub-balances, use the **pin_sub_balance_cleanup** utility. See "pin_sub_balance_cleanup" in *BRM System Administrator's Guide*.

Configuring Time-Stamp Rounding for Cycle Events

You can configure BRM to round time stamps to midnight for balances granted by cycle events. For more information, see "Configuring Time-Stamp Rounding" in *BRM Configuring and Running Billing*.

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
   ```

   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.

Copyright © 2017, 2026, Oracle and/or its affiliates.

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 CM.

   For more information, see "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.

> **Note**
>
> When a charge offer'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 end time stamp is rounded for the purpose of calculating the scale to determine the cycle fee amount to charge.

Configuring Time-Stamp Rounding for Purchase Events

You can configure BRM to round time stamps to midnight for balances granted by purchase events. For more information, see "Configuring Time-Stamp Rounding" in *BRM Configuring and Running Billing*.

To enable this feature, run the **pin_bus_params** utility to change the **TimeStampRoundingForPurchaseGrant** business parameter. See "pin_bus_params" in *BRM Developer's Guide* for more information.

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 Events". 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**:

   ```
   <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.

Copyright © 2017, 2026, Oracle and/or its affiliates.

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*.

Copyright © 2017, 2026, Oracle and/or its affiliates.

26

Configuring Dynamic Quota

Learn how to configure dynamic quota in Pricing Design Center (PDC) to allocate the available quota dynamically for each parallel session of a subscriber based on the rules you configure.

Topics in this document:

- Configuring Dynamic Quota Selectors

Configuring Dynamic Quota

> **Note**
>
> You can use Dynamic Quota Configuration only in Oracle Communications Elastic Charging Engine (ECE).

Dynamic quota allows you to allocate the available quota dynamically for each parallel session of a subscriber based on the rules you configure. This enables the subscribers to run concurrent online charging sessions effortlessly. This also enables you to optimize the network usage effectively. For more information on dynamic quota, see "Managing Dynamic Quotas for Online Sessions" in *ECE Implementing Charging*.

You can define the rules for determining the dynamic quota allocation by configuring dynamic quota selectors in PDC. The rules can reference any of, or a combination of, the event, service, and customer attributes, and profile attribute specifications. You can configure the dynamic quota selector for each service and event combination.

ECE evaluates the rules defined in the dynamic quota selector at run time to derive the quota to be allocated for a session and the quota attributes, such as quota holding time (QHT), volume quota threshold (VQT), and quota validity time (VT). Dynamic quotas change in real time based on the dynamic quota selector rules as ECE grants and redistributes the quotas.

You configure dynamic quota selectors by importing the rules in XML files by using the **ImportExportPricing** utility. You can use the **SampleDynamicQuotaSelector.xml** file in the *PDC_home***/apps/Samples/Examples** directory (where *PDC_home* is the directory in which the PDC software is installed) to create the XML files for configuring dynamic quota selectors. See "Configuring Dynamic Quota Selectors" for more information.

Table 26-1 lists the elements that are supported in the dynamic quota selector XML file, the usage of each element, and a description of how to specify each element in the XML file.

Table 26-1 Elements Supported in the Dynamic Quota Selector XML File

Element	Syntax	Description
dynamicQuotaSelectors	`<dynamicQuotaSelectors>`*`DynamicQuotaSelector`*`</dynamicQuotaSelectors>`	*DynamicQuotaSelector* specifies the root element of the **DynamicQuotaSelector.xml** file.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 26-1 (Cont.) Elements Supported in the Dynamic Quota Selector XML File

Element	Syntax	Description
name	`<name>DynamicQuotaConfiguration</name>`	*DynamicQuotaConfiguration* specifies the dynamic quota selector name. For example, Generic Selector ABC.
description	`<description>DynamicQuotaSelectorDescription</description>`	*DynamicQuotaSelectorDescription* specifies the description of the dynamic quota selector.
priceListName	`<priceListName>PriceListName </priceListName>`	*PriceListName* specifies the price list name. The valid value is **Default**.
applicableToName	`<applicableToName>ApplicableToName</applicableToName>`	*ApplicableToName* specifies the service name. The attributes of this service are used in the rules.
eventSpecName	`<eventSpecName>EventName</eventSpecName>`	*EventName* specifies the name of an event. The attributes of this event are used in the rules.
validityPeriod	`<validFrom>Validfromdate</validFrom>`	*Validfromdate* specifies the date from which the rule is valid. The valid format is *YYYYMMDD*. **Note:** It is mandatory to mention the date from when the rule is valid. If no date is specified, the rule is valid from immediate effect.
applicableToAllChildEvent	`<applicableToAllChildEvent>ChildEventsOption</applicableToAllChildEvent>`	Specifies whether the dynamic quota selector is applicable to the child events, where *ChildEventsOption* is: • **True** if the dynamic quota selector applies to all the child events. • **False** if the dynamic quota selector applies to only the specific event type.
rule	`<rule>` `<ruleName>RuleName</ruleName>` `<ruleOrder>RuleOrder</ruleOrder>` `<configuration>DynamicQuotaConfiguration</configuration>` `<requestedUnits>RequestedUnitsForAllocation</requestedUnits>` `<dynamicQuotaFieldToValueExpression>FieldToValueExpression</dynamicQuotaFieldToValueExpression>` `<dynamicQuotaComplexExpression>ComplexExpression</dynamicQuotaComplexExpression>` `</rule>`	Specifies the rules for allocating dynamic quotas, where: • *RuleName* specifies the rule name. • *RuleOrder* specifies the order in which the rules must be applied. • *DynamicQuotaConfiguration* contains the configurations for dynamic quotas. • *RequestedUnitsForAllocation* contains the requested units for allocating dynamic quotas. • *FieldToValueExpression* contains the attributes used for deriving the quota. • *ComplexExpression* contains the complex conditions for deriving the quota. **Note:** You can add multiple rules in the dynamic quota selector. Each rule in the dynamic quota selector can refer to a different attribute. For example, Rule 1 can use an event attribute, Rule 2 can use a balance expression, and Rule 3 can use a service attribute.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 26-1 (Cont.) Elements Supported in the Dynamic Quota Selector XML File

Element	Syntax	Description
DynamicQuotaConfiguration	`<configuration>` `  <key>`*key*`</key>` `  <value>`*value*`</value>` `  <unit>`*unit*`</unit>` `</configuration>`	Specifies the quota attribute configurations, where: • *key* specifies the quota attribute. The valid values are: **VOLUME_QUOTA_THRESHOLD.** Specifies how long a granted quota can be idle before the reservation is released. **QUOTA_HOLDING_TIME.** Specifies how much of the granted quota must be consumed before a subscriber can request additional quota. This attribute is configured per service, event, and number of granted units. **VALIDITY_TIME.** Specifies whether validity time can be set to a fixed value per service-event combination at runtime. This attribute is independent of the number of units in the granted quota. • *value* specifies the quota attribute value. • *unit* specifies the unit for measurement. The valid values are: **Seconds** **Minutes** **Hours** **Days** **Bytes** **Kbytes** **Mbytes** **Gbytes** **None** **Note:** You must add the *DynamicQuotaConfiguration* element for each valid quota attribute.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 26-1 (Cont.) Elements Supported in the Dynamic Quota Selector XML File

Element	Syntax	Description
RequestedUnitsForAllocation	`<requestedUnits>` `  <fieldName>`*`fieldname`*`</fieldname>` `  <unit>`*`unit`*`</unit>` `  <dynamicQuotaBinaryExpression>`*`dynamicQuotaBinaryExpression`*`</dynamicQuotaBinaryExpression>` `</requestedUnits>` `<requestedUnits>` `  <fieldName>`*`fieldname`*`</fieldname>` `  <unit>`*`unit`*`</unit>` `  <numberExpression>`*`numberExpression`*`</numberExpression>` `</requestedUnits>`	Specifies the details about the requested units, where: • *fieldName* specifies the fully qualified requested attribute field name. This is specified in the event specification. For example, *EventDelayedSessionTelcoGsm.REQUESTED_UNITS.INPUT_VOLUME*. • *unit* specifies the unit for measurement. The valid values are: **Seconds** **Minutes** **Hours** **Days** **Bytes** **Kbytes** **Mbytes** **Gbytes** **None** • *dynamicQuotaBinaryExpression* specifies the condition for deriving the quota. • *numberExpression* specifies a valid number; for example, 10.0.
dynamicQuotaBinaryExpression	`<dynamicQuotaBinaryExpression>` ` <leftOperand>` `  <numberExpression>`*`numberExpression`*`</numberExpression>` `  <balanceExpression>`*`balanceExpression`*`</balanceExpression>` ` </leftOperand>` ` <rightOperand>` `  <balanceExpression>`*`balanceExpression`*`</balanceExpression>` `  <numberExpression>`*`numberExpression`*`</numberExpression>` ` </rightOperand>` ` <dynamicQuotaBinaryOperator>`*`Operator`*`</dynamicQuotaBinaryOperator>` `</dynamicQuotaBinaryExpression>`	Specifies the condition for deriving the quota, where: • *numberExpression* specifies a valid number; for example, 10.0. • *balanceExpression* specifies the currency or noncurrency balance element defined in PDC. • *Operator* specifies the arithmetic operators. The valid values are: **Add** **Subtract** **Multiply** **Divide**

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 26-1 (Cont.) Elements Supported in the Dynamic Quota Selector XML File

Element	Syntax	Description
FieldToValueExpression	`<dynamicQuotaFieldToValueExpression>` `<separator>`*`Delimiter`*`</separator>` `<operation>`*`Operator`*`</operation>` `<fieldName>`*`FieldName`*`</fieldName>` `<fieldKind>`*`FieldKind`*`</fieldKind>` `<fieldValue>`*`FieldValue`*`</fieldValue>` `</dynamicQuotaFieldToValueExpression>`	Specifies the attributes used for deriving the quota, where: • *Delimiter* specifies the character that is used to separate the field values. **Note:** This is applicable only when the *operator* is **IN_LIST**. • *Operator* specifies the operator to be used in the expression. The valid values are: **EQUAL_TO** **NOT_EQUAL_TO** **GREATER_THAN** **GREATER_THAN_EQUAL** **LESS_THAN** **LESS_THAN_EQUAL** **IN_LIST** **CONTAINS** • *FieldName* specifies the fully qualified field name. For example, EventSessionTelcoGsm.REQUESTED_UNITS.NUMBER_OF_UNITS. • *FieldKind* specifies the type of attributes. The valid values are: **PRODUCT_SPEC_FIELD** **EVENT_SPEC_FIELD** **CUSTOMER_SPEC_FIELD** **PROFILE_SPEC_FIELD** • *FieldValue* specifies the field value that is used to track the balance impacts. **Note:** You can add multiple *FieldToValueExpression* elements in a rule to derive the quota based on a combination of attributes.
ComplexExpression	`<dynamicQuotaComplexExpression>` `<operator>`*`Operator`*`</operator>` `<value>`*`Value`*`</value>` `<dynamicQuotaBinaryExpression>`*`dynamicQuotaBinaryExpression`*`</dynamicQuotaBinaryExpression>` `</dynamicQuotaComplexExpression>`	Specifies complex conditions for deriving the quota, where: • *Operator* specifies the operator to be used in the expression. The valid values are: **EQUAL_TO** **NOT_EQUAL_TO** **GREATER_THAN** **GREATER_THAN_EQUAL** **LESS_THAN** **LESS_THAN_EQUAL** • *Value* is a numeric value. For example, 0.004. • *dynamicQuotaBinaryExpression* specifies the condition for deriving the quota.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Configuring Dynamic Quota Selectors

To configure a dynamic quota selector:

1. Make a copy of the *PDC_home*/**apps/Samples/Examples/Sample_DynamicQuotaSelector.xml** file to customize.
2. Open the file in a text editor or XML editor.
3. Add or modify the elements as required. See Table 26-1 for the usage and description of each element.
4. Save and close the file.

> **Note**
>
> You can save the file with a different name.

5. Go to the directory where you have saved the file.
6. Import the dynamic quota selector from the customized XML file into the PDC database by running the following command:

   ```
   ImportExportPricing -import -config FileOrDirectoryName
   ```

   where *FileOrDirectoryName* is the name of the customized XML file, such as **Dynamic_QuotaSelector.xml**. Alternatively, enter the path to the directory containing multiple XML files to import.

   The dynamic quota selector is imported into the PDC database and published to ECE.

Copyright © 2017, 2026, Oracle and/or its affiliates.

27

About Folds

Learn how to use folds in Oracle Communications Billing and Revenue Management (BRM) product offerings.

Topics in this document:

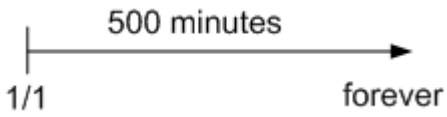
- About Folds
- About Applying Folds Only For Account-Level Products

About Folds

When a fold event occurs, BRM checks whether the balance is valid or expired. The balance is folded if the sub-balance is expired or if the sub-balance is always valid. For example:

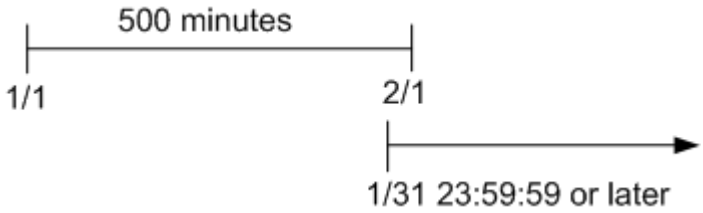
- 500 minutes are valid forever, so the balance can be folded at any time as shown in Figure 27-1.

Figure 27-1 Minutes Valid Forever



- 500 minutes that expire on February 1 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 as shown in Figure 27-2.

Figure 27-2 500 Minutes Expiring on February 1st



Copyright © 2017, 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 balances before final billing is run. However, free balances are granted at the beginning of each cycle before the fold event occurs. If the newly granted balances do not have a configured validity period (that is, their validity period never ends), the fold event will remove those balances.
>
> When you use fold events to remove unused balances and you also use delayed billing, you should always configure a validity period for the granted balances. This prevents fold events from removing balances that are granted at the beginning of a cycle before final billing is run.

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.

Copyright © 2017, 2026, Oracle and/or its affiliates.

28

Applying Credit Limits to Balances

Learn how to implement and manage credit limits in Oracle Communications Billing and Revenue Management (BRM) product offerings.

Topics in this document:

- About Credit Limits
- About Credit Thresholds and Floors
- Handling Credit Limit Conflicts
- Allowing Customers to Exceed Their Credit Limit

See also "Managing Balances During Charging".

About Credit Limits

A credit limit is the maximum amount of a balance element that can accumulate in a balance group. When a credit limit is reached, businesses typically deny customers access to the services associated with the balance group. For example, you might set a credit limit of $100 for a telephony package and deny service when customers who reach that limit try to place a call.

You can also configure charges to allow customers to exceed their credit limits. See "Allowing Customers to Exceed Their Credit Limit".

You set credit limits for balance elements in packages.

About Credit Thresholds and Floors

You can use credit thresholds to notify customers when they are approaching the credit limit of a balance. A credit threshold specifies the balance total that triggers a notification to the customer. You can specify the threshold in the following 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 balance).

The *credit floor* is the starting point for a credit threshold and is the lowest number that the balance can be (that is, the number that represents no use of the balance).

For currency balances, the credit floor is 0.

For noncurrency balance elements, such as prepaid hours, you must specify a credit floor. You can use a negative number for the floor. For example, suppose you give 100 prepaid hours and set the credit limit to 0. When the credit limit is reached, the customer has no hours remaining and cannot use the service. To notify the customer when only 10 hours remain, set the credit threshold and floor as follows:

- Set the credit floor to -100. This number indicates none of the balance has been used.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- 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).

The credit threshold can be triggered both when a balance is increasing and when it is decreasing. 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. If the threshold is crossed when the balance is decreasing, service could be restored.

Handling Credit Limit Conflicts

There might be occasions when customers purchase packages with credit limits that conflict with the credit limit set on their account or another service. There are two ways to handle credit limit conflicts:

- When you create your product offerings, if two services have different credit limits, create new balance groups for the services. This enables credit limits for each service to be set and tracked independently. See "Tracking Balances in Balance Groups".
- Specify which credit limit takes precedence when a new credit limit conflicts with an existing credit limit. You do this by setting the **credit_limit_conflict** entry in the Connection Manager (CM) configuration file.

To specify credit limit precedence:

1. Open the CM configuration file (*BRM_home***/sys/cm/pin.conf**).
2. Add the **credit_limit_conflict** entry:

   ```
   -fm_bill credit_limit_conflict value
   ```

   where *value* is one of these values:

   - **replace** to use the new credit limit.
   - **ignore** to ignore the new credit limit and keep the existing credit limit.
   - **add** to add the new credit limit to the existing credit limit and create a new limit.
   - **minimum** to use the credit limit that specifies the smaller amount.
   - **maximum** to use the credit limit that specifies the greater amount.

   If this entry is not present, the new credit limit is used by default.
3. Stop and restart the CM.

Allowing Customers to Exceed Their Credit Limit

When configuring properties for quantity ranges in charges, you can specify what happens to subscriptions when customers exceed their credit limits. You can do the following:

If you do not want to allow customers to exceed their credit limit, you can specify that:

- The subscription succeeds and the resources are prorated according to the available balance. For example, if the subscription grants 2GB of data and costs $10, but the customer only has $8 in their account, the $8 would be used and the customer would be granted 1.6GB of data.
  In the PDC UI, this is the **Enforce Credit Limit** option.
- 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.

Copyright © 2017, 2026, Oracle and/or its affiliates.

In the PDC UI, this is the **Rental Failure** option.

You configure the maximum number of retries in the **max_retry_count** subscription business parameter. The default is **0**, which sends the failure notification immediately without retrying. See "Using Event Notification" in *BRM Developer's Guide* for information about notifications and "Configuring the Maximum Number of Subscription Retries" for information about the maximum number of retries.

The current number of retries for the subscription is recorded in the PIN_FLD_RETRY_COUNT field of the **/purchased_product** object.

If you create charge offers in XML using the ImportExportPricing utility, you can also configure the maximum retries and their interval at the offer level. See "Enforcing or Exceeding Credit Limits" for more information. If nothing is configured for the offer, the business parameter is used.

If you want to allow customers to exceed their credit limit, you can specify that:

- The subscription succeeds and all available balance is used. The remaining amount is recorded as a negative account balance to be paid at the next top up for prepaid customers or next bill for postpaid customers.
  In the PDC UI, this is the **Do not enforce credit limit** option. You typically use this option for postpaid customers.
- 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 next bill for postpaid customers.
  In the PDC UI, this is the **Deduct Rental** option. You typically use this option for prepaid customers.
- 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.
  In the PDC UI, this is the **Loan** option.

  See "About Loans" in *BRM Configuring and Collecting Payments* for more information about loans.
- 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 next bill for postpaid customers.
  In the PDC UI, this is the **Outstanding Amount** option.
- 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. Services are still available.
  In the PDC UI, this is the **Skip Cycle** option.

If you create charges using the ImportExportPricing utility, you have additional options for exceeding credit limits. See "Enforcing or Exceeding Credit Limits" for more information.

Configuring the Maximum Number of Subscription Retries

When a customer's subscription fails due to insufficient credit, the subscription can be retried a configured number of times before sending a failure notification to an external system.

The maximum number of retries is configured in the **MaxRetryCount** field in the business parameters for subscriptions. By default, this is set to **0**, which sends the notification immediately without retrying.

To configure the maximum number of subscription retries:

Copyright © 2017, 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 subscription instance of the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsSubscription bus_params_subscription.xml
   ```

   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 **MaxRetryCount** to the maximum number of subscription retries allowed.

   ```
   <MaxRetryCount>value</MaxRetryCount>
   ```

   The default value is **0**.

   > ⚠ **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 CM.

   For more information, see "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.

If you create charge offers in XML using the ImportExportPricing utility, you can also configure the maximum retries and their interval at the offer level. See "Enforcing or Exceeding Credit Limits" for more information. If nothing is configured for the offer, the business parameter is used.

Copyright © 2017, 2026, Oracle and/or its affiliates.

29

About Rollovers

Learn how rollovers are managed in Oracle Communications Billing and Revenue Management (BRM) product offerings.

Topics in this document:

- About Rollovers
- When Rollover Events Occur
- About Rolling Over Noncurrency Balances during Package Transition
- About Rolling Over Balances That Expire in Midcycle
- Prorating Rollover Balances
- Rollover Example

About Rollovers

A rollover is a type of charge that credits unused balances at the end of a billing cycle to the account balance for the next cycle. For example, minutes are often rolled over from one month to the next. Balances from rollovers are different from granted balances, which are valid between two specific dates.

You define rollovers in charge offers in Oracle Communications Pricing Design Center (PDC). When defining rollovers, you specify the following:

- The period in which the rollover is valid. You can specify start and end time dates.
- The balance to roll over.
- The type of cycle to which the rollover applies (for example, monthly).
- The maximum number of cycles that a balance can be rolled over to.
- The maximum amount that can be rolled over in a cycle.
- The maximum value for a rollover balance.
- A G/L ID for the rollover charge.

BRM maintains each rollover balance as a sub-balance.

BRM validates whether a rollover sub-balance can be rolled over by checking the rules that govern the 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 rollover sub-balance:

- **If the full amount is eligible for rollover**, BRM does one of the following:
  - If the existing rollover sub-balance has the same properties as the new rollover sub-balance (such as the balance type, rollover rules, valid-from and valid-to dates, sub-balance contributors, and so on), BRM adds the rolled over amount to the existing rollover sub-balance.

Copyright © 2017, 2026, Oracle and/or its affiliates.

  - If the existing rollover sub-balance has different properties than the new rollover sub-balance, BRM 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.

- **If only a portion of the balance is eligible for rollover**, the amount in the rollover sub-balance is divided into a nonrollover sub-balance and a rollover sub-balance. The nonrollover sub-balance has the same valid-to date as the original rollover sub-balance Its 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 balances are available for the customer to use in the new cycle.
- **If none of the balance is eligible for rollover**, the rollover sub-balance is not changed. This condition occurs when the rollover sub-balance has already been rolled over the maximum number of times allowed. Amounts in this rollover sub-balance are available only for late-arriving usage events.

> **Note**
>
> BRM memory limits might limit the number of months you can roll over a rollover sub-balance. The number of rollover sub-balances that BRM must maintain varies according to the number of rollover cycles allowed. You set the maximum number of rollover cycles allowed when configuring the rollover balance impact in PDC.

When Rollover Events Occur

Rollover sub-balances can be rolled over to another cycle as soon as they surpass the valid-to date (expire). Most rollover sub-balances expire on the account's billing day of month (DOM). However, some rollover sub-balances may expire in the middle of a cycle due to a charge offer or service cancellation or due to flexible cycles.

Because many BRM features depend on rollover sub-balances being rolled over the day after they expire, BRM rolls over balances at the following times:

- **At the end of the billing cycle**: BRM automatically rolls over all eligible rollover sub-balances to the next cycle as part of the billing process. This catches all rollover 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 rollover sub-balances that have expired but have not been rolled over to the next cycle. This catches any rollover sub-balances that expire in the middle of the cycle.

About Rolling Over Noncurrency Balances during Package Transition

When your customers transition from one package to another, you can specify that noncurrency balances be rolled over from one package to another if both packages are in at least one package list together.

To specify that noncurrency balances rollover between packages during package transition, you must perform these tasks:

Copyright © 2017, 2026, Oracle and/or its affiliates.

- 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 package lists, ensure that packages between which you want to allow rollovers are in at least one package list together.

These general rules apply to controlled rollovers:

- Controlled rollovers are not affected by the charge offer cycle rollover settings.
- A controlled rollover does not count as a rollover sub-balance 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 noncurrency balances that have been prorated during the package transition are rolled over to the next package.

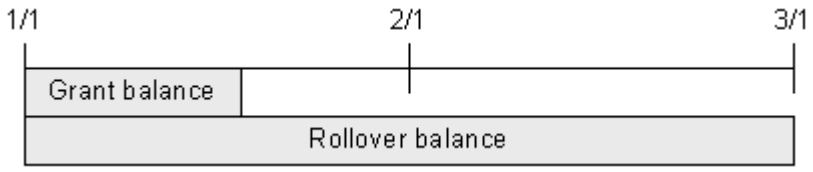
About Rolling Over Balances That Expire in Midcycle

A balance can expire in the middle of a cycle: for example, when the balance is valid for only minutes, hours, or days or when the balance is valid for one or more months and starts in the middle of a month.

If a balance's validity period does not end when the cycle ends, the rollover sub-balance is valid for the entire cycle in which the balance is granted and for the whole of the next cycle.

In Figure 29-1, a monthly cycle event grants 60 minutes that are valid for two weeks from the grant date. The minutes are granted on January 1. On January 14, when the balance of minutes expires, any remaining balances that can be rolled over are added to a new rollover sub-balance, which is valid from January 1 to March 1:

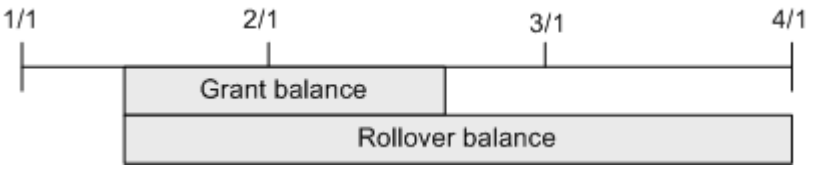
Figure 29-1 Grant and Rollover Balance Date Validity Example 1



The same rollover period applies when the validity period of the minutes ends after the cycle in which they are granted. In Figure 29-2, a monthly cycle event grants 300 minutes that are valid for six weeks from the grant date. The minutes are granted on January 1. On February 11, when the balance expires, any remaining balances that can be rolled over are added to a new rollover sub-balance, which is valid from January 1 to March 1:

If the balance's validity period does not start at the beginning of the last cycle, the balances are rolled over for one entire cycle.

Figure 29-2 Grant and Rollover Balance Date Validity Example 2



Copyright © 2017, 2026, Oracle and/or its affiliates.

Prorating Rollover Balances

Customers typically purchase and cancel charge offers at some point in the middle of a cycle. When you set up a rollover in PDC, you can specify whether rollover sub-balances are prorated for the first and last cycles based on the number of days in the cycles that the charge offer is owned.

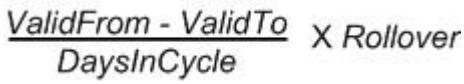
Table 29-1 describes the rollover proration options:

Table 29-1 Rollover Proration Options

Proration Option	Description
Roll over entire amount	Roll over the available monthly rollover balances.
No rollover	Do not roll over any available monthly rollover balances.
Prorate rollover amount	Calculate the rollover balances 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 29-3 to calculate the amount to roll over:

Figure 29-3 Calculation for Amount to Roll Over



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 balance, such as 100 minutes.

For example, suppose:

- A customer buys a charge offer on January 15 that grants 500 minutes of use during each cycle.
- A rollover is set up so that up to 200 unused 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 29-2 describes how the rollover is handled, depending on the purchase midcycle proration setting:

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 29-2 Impact of Proration Options

Proration Option	Amount Rolled Over into February
Roll over entire amount	200 minutes are rolled over from January into February. In February, the customer has 700 minutes (200 rolled over plus the 500 minutes February grant).
No rollover	No minutes are rolled over from January into February. In February, the customer has only the February grant of 500 minutes.
Prorate rollover amount	Approximately 110 minutes are rolled over from January into February: (17/31) * 200 = 109.67 In February, the customer has about 610 minutes (110 rolled over plus the 500 minutes February grant).

A similar set of results applies to the last-month proration calculation specified in the cancel midcycle proration setting.

Rollover Example

In this example, the customer's charge offer includes:

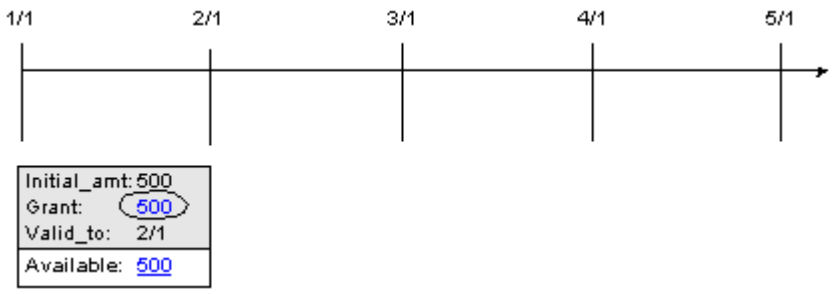
- 500 minutes granted at the beginning of each usage cycle.
- A rollover specifying that:
  - Up to 100 unused minutes from each month can be rolled over.
  - The maximum number of rollover cycles is 2.
  - A limit of 150 *total* rollover minutes from previous months can be rolled over into a new month.

Other assumptions:

- Cycles are monthly and start on the first day of each calendar month.
- The charge offer with the rollover is valid starting January 1.
- The customer consumes no minutes until March.

1. On January 1, the customer is granted 500 minutes for use during January as shown in Figure 29-4. The minutes are maintained in a sub-balance.

Figure 29-4 Initial 500 Minutes Grant in January

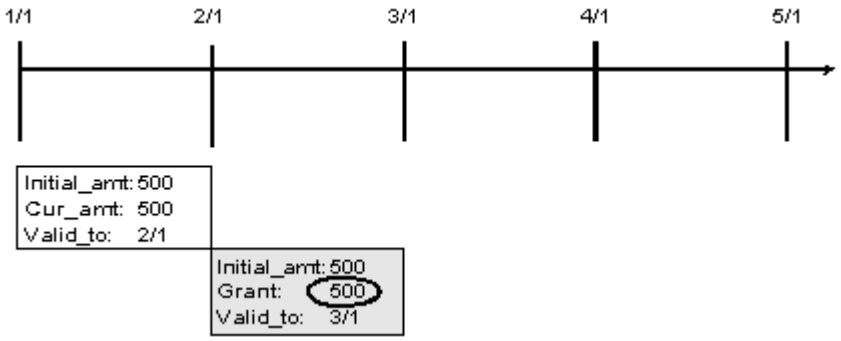


2. On February 1:

Copyright © 2017, 2026, Oracle and/or its affiliates.

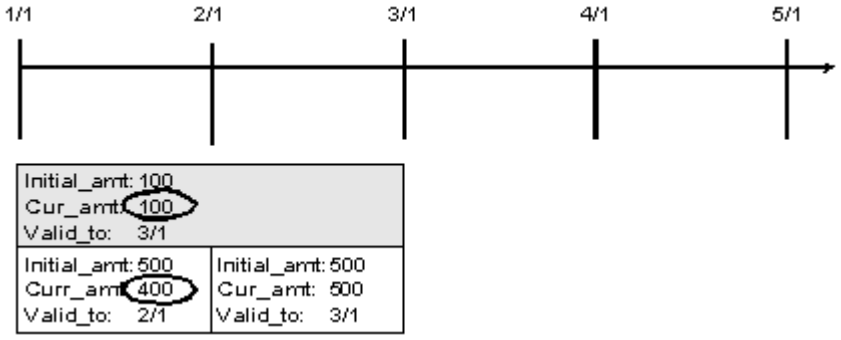
- The cycle forward event creates a new sub-balance to track the grant of 500 minutes for use during February as shown in Figure 29-5.

Figure 29-5 February Minutes Grant



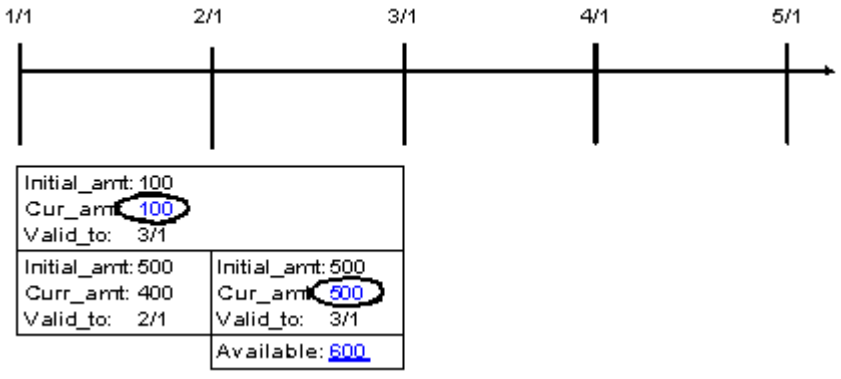
- The cycle rollover event creates a new sub-balance for tracking unused January minutes that roll over into February. 100 minutes are put in the new sub-balance, and they are valid until March 1. The original sub-balance for January is decremented 100 minutes, leaving 400 minutes available for late-arriving January events as shown in Figure 29-6.

Figure 29-6 January Minutes Rollover



The customer now has 600 minutes (500 granted February 1 plus 100 rolled over from January) available for use during February as shown in Figure 29-7.

Figure 29-7 February Total Minutes Available

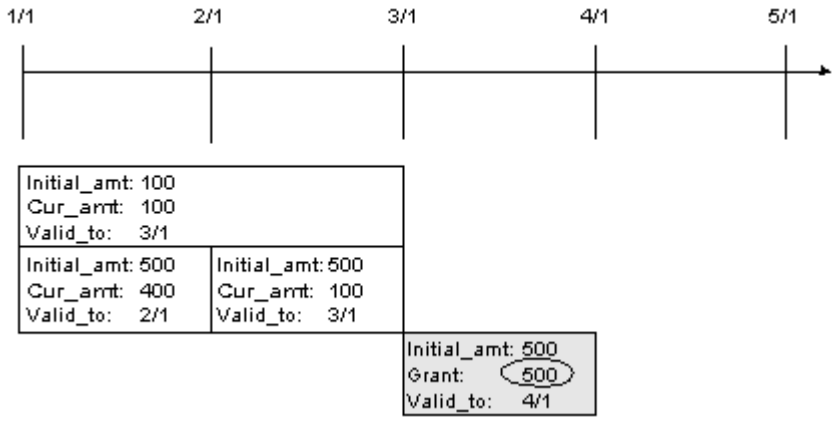


Copyright © 2017, 2026, Oracle and/or its affiliates.

3. On March 1:

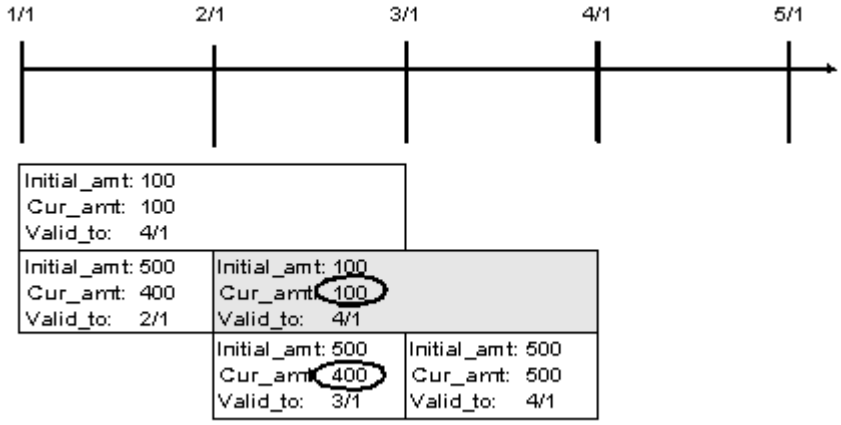
   - The cycle forward event creates a new sub-balance to track the grant of 500 minutes for use during March as shown in Figure 29-8.

Figure 29-8 March Minutes Grant



   - The cycle rollover event creates a new sub-balance for tracking unused February minutes that roll over into March as shown in Figure 29-9. 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 29-9 February Minutes Rollover

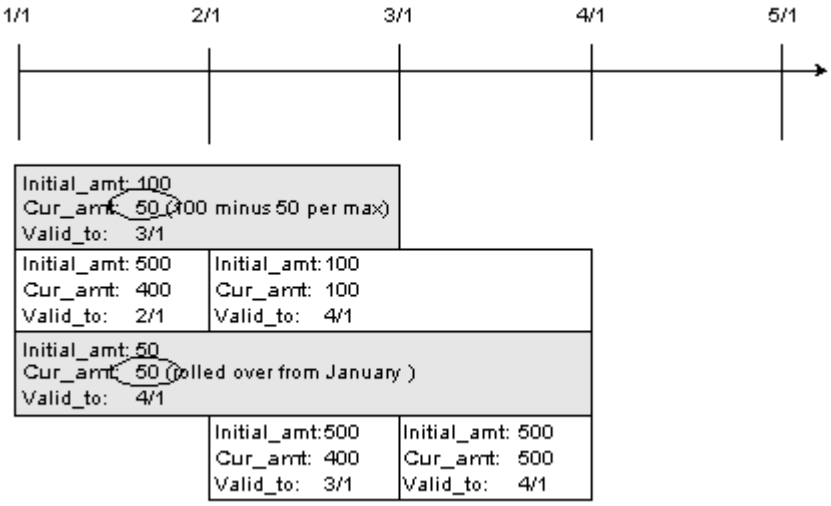


   - The cycle rollover event divides the January rollover minutes into two sub-balances to enforce the rule that only 150 *total* rollover minutes for a balance can carry forward into a new month. Because 100 free February minutes are already rolled over, only 50 minutes can be rolled over from the January 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 balances for each month are shown in Figure 29-10.

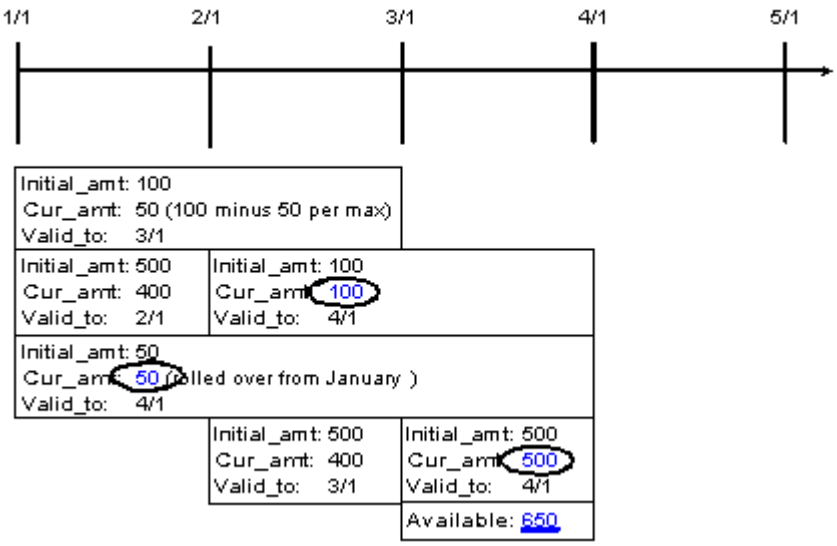
Copyright © 2017, 2026, Oracle and/or its affiliates.

Figure 29-10 Remaining January Minutes Rollover



The customer now has 650 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 29-11.

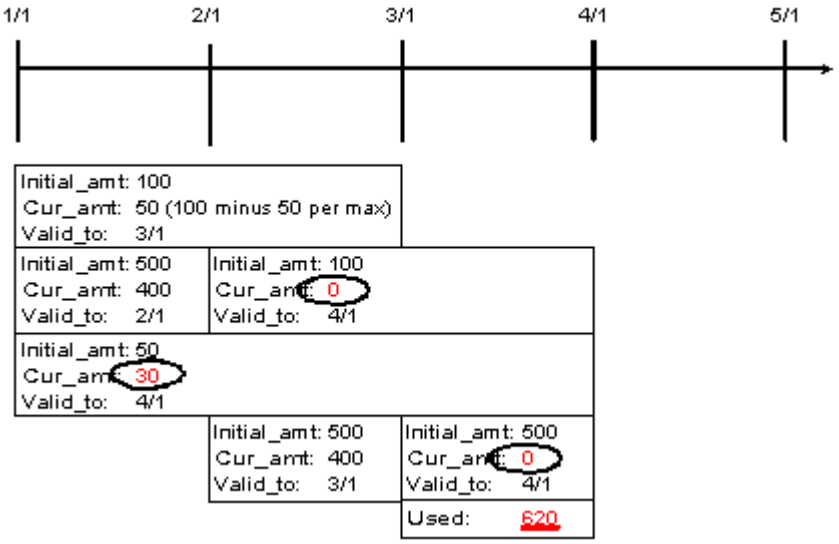
Figure 29-11 March Total Minutes Available



4. During March, the customer consumes 620 minutes as shown in Figure 29-12. Sub-balances are used starting with the newest sub-balance, as indicated by the sub-balance valid-from dates:
   - All 500 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 © 2017, 2026, Oracle and/or its affiliates.

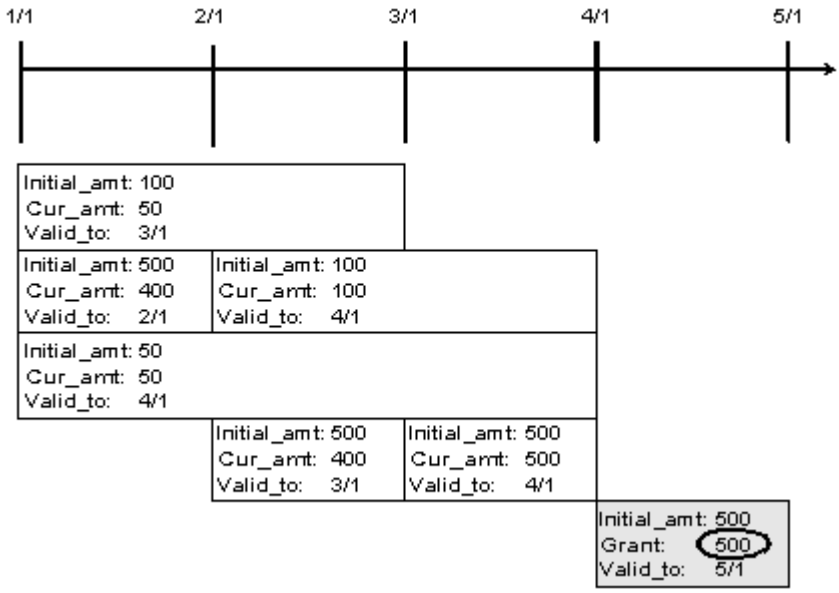
Figure 29-12 March Minutes Usage



5. On April 1, the cycle forward event creates a new sub-balance to track the grant of 500 minutes for use during April as shown in Figure 29-13.

   No minutes roll over into April from the March and February rollovers because these balances were consumed in March. The 30 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 29-13 April Minutes Grant



The customer has only 500 minutes available for April.

Copyright © 2017, 2026, Oracle and/or its affiliates.

> **Note**
>
> If two charge offers contributing to the same balance group have rollover charges configured for the same balance with the same rollover frequency, either of the charge offers can be used to roll over the balances. In this case, rollover results may vary depending on the charge offer that is selected first.

Copyright © 2017, 2026, Oracle and/or its affiliates.

30

Configuring Balance Impact Rounding

Learn how to configure balance impact rounding rules in Oracle Communications Billing and Revenue Management (BRM) product offerings.

Topics in this document:

- About Balance Impact Rounding
- Configuring Balance Impact Rounding

About Balance Impact Rounding

BRM enables you to create various rounding rules for different balance elements, 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.

About Rounding Rules

You can configure balance impact 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.

  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 30-1 show how a US dollars balance element and purchase event combination can be rounded in various ways for various processes.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 30-1 Currency Balance Element and Event Type

For a US Dollars Balance Element	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 balance element, 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.

How BRM Applies Rounding

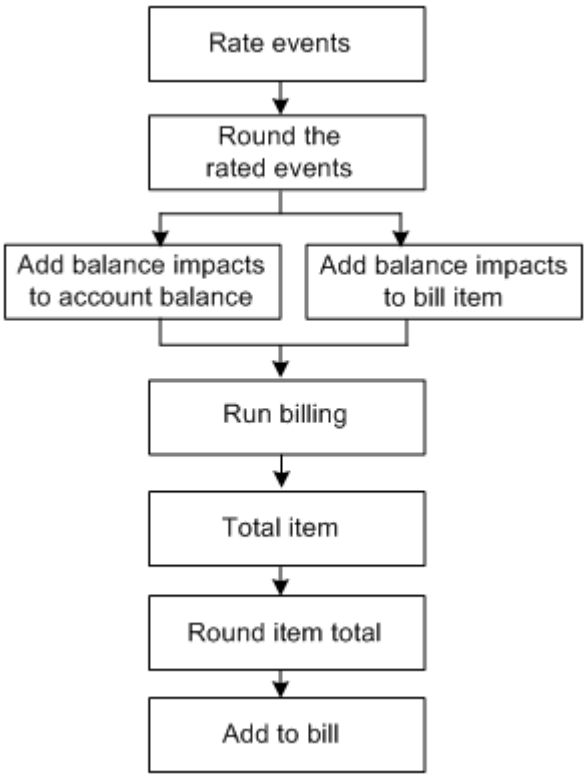
Rating, discounting, and taxation produce balance impacts, which are rounded and applied to the customer's account balance. The balance impacts are rounded to the scale and mode configured for the event, balance element, and process combination.

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 30-1 shows the general process of balance impact rounding.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Figure 30-1 BRM Balance Impact 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 package 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 an offer 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, 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 rounded fee, the tax is rounded, and a balance impact is generated with the rounded amount.
5. **Billing discounts**. When you run billing, if a billing-time discount applies, it is applied to the total of the rounded usage items, 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

Copyright © 2017, 2026, Oracle and/or its affiliates.

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 balance element and cycle forward event combination. Rollovers are 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 a *natural scale*: that is, the scale commonly used in the marketplace for that balance element. 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 balance element. However, the events associated with billed items still retain their pre-billing scale.

If any operation is performed on billed items: for example, event and item 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. 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 30-2, 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 30-2 Rounding for Specific Event Types

Balance Element	Event Type	Process	Rounding Scale	Rounding Mode
US dollars	**/event/session**	Rating	6	Down
US dollars	***** (all events)	Taxation	2	Nearest

About Rounding Counter Balances for Discounting

For discount rounding, the rounding configurations are used to round only the balance impacts of discounting, not input balances, such as counters. To configure counter rounding, see "Configuring Rounding Rules for Counter Balances".

Copyright © 2017, 2026, Oracle and/or its affiliates.

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 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.

- 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 nonalternative counterparts (ROUND_FLOOR and ROUND_DOWN), except that they compensate for possible loss of

Copyright © 2017, 2026, Oracle and/or its affiliates.

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 nonalternative 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 30-3.

Table 30-3 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

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 nonalternative 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 nonalternative rounding modes.

Copyright © 2017, 2026, Oracle and/or its affiliates.

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 Balance Impact Rounding

To set up rounding, perform the following tasks:

- Configuring Rounding Rules for Counter Balances
- Configuring to Record Rounding Differences in the G/L
- Configuring to Record Rounding Differences in the G/L
- Rounding Examples

About Configuring Rounding Rules

You configure rounding for balance element and event type combinations (for example, US dollars and **/event/session/telco/gsm** events). For each balance element and event combination, you specify a rounding scale, a rounding mode, and the process for which this rule applies.

> **Note**
>
> You do not need to configure rounding for all of the processes. However, you must configure rounding for every balance element and event type that has a balance impact.

About Rounding Mode Values in the BRM API

The rounding modes you specify have corresponding modes in the BRM API. 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.

Table 30-4 shows the BEID rounding modes and the corresponding rounding parameters for the decimal data type functions.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 30-4 BEID Rounding Modes

BEID Rounding Mode (Specified in pin_bill.h)	Rounding Mode Parameter for pbo_decimal Functions
0: PIN_BEID_ROUND_NEAREST	**5**: ROUND_HALF_UP
1: PIN_BEID_ROUND_UP	**1**: ROUND_UP
2: PIN_BEID_ROUND_DOWN	**2**: ROUND_DOWN
3: PIN_BEID_ROUND_EVEN	**7**: ROUND_HALF_EVEN
4: PIN_BEID_ROUND_FLOOR	**4**: ROUND_FLOOR
5: PIN_BEID_ROUND_FLOOR_ALT	**8**: ROUND_FLOOR_ALT
6: PIN_BEID_ROUND_DOWN_ALT	**9**: ROUND_DOWN_ALT
N/A	**3**: ROUND_CEILING
N/A	**6**: ROUND_HALF_DOWN
N/A	**10**: ROUND_UNNECESSARY

Configuring Rounding Rules for Counter Balances

You use Oracle Communications Pricing Design Center (PDC) to configure rounding rules for counter balances. When you configure discounts, use discount expressions to specify how to round the balance. Use the following discount expression syntax:

`round(`*`expression, rounding_scale, rounding_mode`*`)`

where *expression* defines the balance to round. This can be any discount expression. To round a counter balance, use the **Balance** expression. For more information, see the discussion about configuring discounts in PDC Help.

For example, to round a counter balance down to two decimal places for balance element ID 100099, use the following expression:

```
round( Balance(1000099), 2, ROUND_DOWN )
```

About Rounding Modes for Discount Expressions

Rounding modes for discount expressions have slightly different names. Table 30-5 lists the rounding mode values you can specify.

Table 30-5 Rounding Modes for Discount Expressions

Discount Expression Rounding Mode	Rounding Mode
ROUND_PLAIN	Nearest
ROUND_UP	Up
ROUND_DOWN	Down
ROUND_BANKERS	Even

For a definition of what these modes represent, see "About Rounding Modes".

Copyright © 2017, 2026, Oracle and/or its affiliates.

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:

- 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 "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
)
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

> **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:

   load_pin_glid *pin_glid_file*

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 "Creating a Localized Version of BRM" in *BRM Developer's Guide*.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Configuring BRM to Record Rounding Differences

By default, rounding differences are not recorded in G/L reports. You can enable recording the difference between rounded bill items and the rounded total by running the **pin_bus_params** utility to change the **GenerateJournalEpsilon** business parameter. For information about this utility, see *BRM Developer's Guide*.

To enable BRM to record rounding differences:

1. Go to *BRM_home*/**sys/data/config**.
2. Create an XML file from the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsBilling bus_params_billing.xml
   ```

3. In the file, change **disabled** to **enabled**:

   ```
   <GenerateJournalEpsilon>enabled</GenerateJournalEpsilon>
   ```

4. Save the file as **bus_params_billing.xml**.
5. Load the XML file into the BRM database:

   ```
   pin_bus_params bus_params_billing.xml
   ```

6. Stop and restart the CM.

Rounding Examples

Rounding is performed after rating, discounting, taxation, and A/R actions such as billing and adjustments.

Table 30-6 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:

- 2 for purchase events, taxation, and A/R
- 5 for rating and discounting

Table 30-6 Rounding Examples

Action/Process	Calculated Charge	Rounded Charge and Balance Impact	Account Balance
Purchase package 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

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 30-6 (Cont.) Rounding Examples

Action/Process	Calculated Charge	Rounded Charge and Balance Impact	Account Balance
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 30-7 shows some rounding results of the ROUND_DOWN_ALT and ROUND_FLOOR_ALT modes as compared to their nonalternative rounding counterparts (ROUND_DOWN and ROUND_FLOOR) for various decimal values and rounding scales.

Table 30-7 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
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

Copyright © 2017, 2026, Oracle and/or its affiliates.

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 30-8 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 30-8 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.

> **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.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Part V
XML Pricing Interface for PDC

This part describes how to use the Pricing Design Center (PDC) XML Pricing Interface to create product offerings.

This part contains the following chapters:

- Importing and Exporting Pricing and Setup Components
- XML Examples of Creating Pricing Components
- PDC XML Tags

Copyright © 2017, 2026, Oracle and/or its affiliates.

31

Importing and Exporting Pricing and Setup Components

Learn how to import and export pricing and setup components in Oracle Communications Billing and Revenue Management (BRM) by using the **ImportExportPricing** utility.

Topics in this document:

- About Using the XML Pricing Interface to Create Product Offerings
- About the ImportExportPricing Utility
- Exporting Pricing and Setup Components from PDC
- Importing Pricing and Setup Components
- Listing Pricing or Setup Components
- Publishing Setup Components
- Deleting Setup Components

> **Note**
>
> You can also use the PDC UI to import and export pricing and setup components. For more information, see "Exporting and Importing Changesets" in *PDC Online Help*.

About Using the XML Pricing Interface to Create Product Offerings

You can create and modify product offerings in an XML file and then use the **ImportExportPricing** command-line utility or the PDC UI to do the following:

- Import all or some product offerings configured in an XML file into the PDC database.

  The utility creates any new pricing and setup components and modifies any changed components in the PDC database.

- Export all or some product offerings from the PDC database into an XML file for editing.

  If you export the data into an existing XML file, the utility overwrites the entire XML file.

You can also use XML files to move product offerings from one PDC database to another PDC database.

To create or modify product offerings in an XML file, you use a text editor or an XML editor. You define the pricing and setup components that you want to create by adding XML tags to the file. See "PDC XML Tags" for information about the required and optional XML tags. You can add the XML tags to:

- One consolidated XML file containing all of the pricing or setup components

Copyright © 2017, 2026, Oracle and/or its affiliates.

- Separate XML files for each type of pricing or setup component

For information about the PDC XML tags that you use to create pricing and setup components, see "PDC XML Tags".

The XML files that you create must conform to the format detailed in the XSD files for pricing or setup components. See "About the XSD Files" for more information.

Some pricing components have dependencies on other pricing and setup components. Before loading such pricing components, ensure that the prerequisite data is available in the PDC database. See Table 31-1 for more information.

If you create XML files for specific pricing or setup components, you must load the XML files in the order based on the dependencies. See "About Dependencies between Pricing and Setup Components" for more information.

About the ImportExportPricing Utility

The **ImportExportPricing** utility is a command-line interface for importing pricing and setup components to and for exporting pricing and setup components from the PDC database by using XML files.

ImportExportPricing is a role-based utility that authenticates and authorizes users based on their role. A user's role determines what tasks they can perform and what information they can view.

Users of this utility can have the following roles:

- **Pricing Design Admin**: Users with this role can import and export all pricing and setup components.
- **Pricing Reviewer**: Users with this role can only export all pricing and setup components.
- **Pricing Analyst**: Users with this role can import only pricing components and cannot import setup components and profile data. Users can export all pricing and setup components.

ImportExportPricing allows you to filter the data that you want to import to or export from PDC. For example, you can export all pricing and setup components from PDC or specify specific pricing and setup components to export.

You can use **ImportExportPricing** to:

- Export pricing and setup components from the PDC database to an XML file. See "Exporting Pricing and Setup Components from PDC" for more information.
- Import pricing and setup components defined in an XML file into the PDC database. See "Importing Pricing and Setup Components" for more information.
- List pricing or setup components available in the PDC database. See "Listing Pricing or Setup Components" for more information.
- Delete setup components from the PDC database if it is not prerequisite data for any other component. See "Deleting Setup Components" for more information.
- Check the current status of one or more changesets after an import or publish operation. Use **changeSetStatus** mode to verify the overall changeset status and the status reported by each target engine. See "Changeset Status Mode" for more information.

Copyright © 2017, 2026, Oracle and/or its affiliates.

About Dependencies between Pricing and Setup Components

Before importing a pricing component, you must import the setup components that the pricing component references. You must ensure that the setup components referenced by the pricing components are either in the XML file or are already imported into PDC.

For example, to import a charge, you must ensure that the balance elements that it references already exist in PDC.

This section assumes that the following prerequisite data is already available in PDC:

- Service
- Event
- Account Attributes
- Service-Event Map
- Profiles

Table 31-1 lists the prerequisites for importing the pricing components into PDC. Some of the prerequisites are optional depending on your business functions.

Table 31-1 Prerequisite Data for Importing Pricing Components

Pricing Component	Prerequisite Pricing Component	Prerequisite Setup Component
Discount Filter	None	None
Time Model	None	None
APN Selector	None	Impact Categories and Zone Models
Pricing	None	RUMs, Balance Elements, G/L IDs, and Tax Codes
USC Selector	None	Impact Categories and Zone Models
Tax Selector	None	Tax Codes
Tax Exemption Selector	None	Tax Exemption Codes
Discount Trigger	None	Balance Elements
Generic Selector	None	Custom Rules and Profile Attribute Specification
Price Selector	Pricing	None
Charge	Time Models, Pricing, USC Selectors, APN Selectors, and Price Selectors	RUMs, Balance Elements, Tax Codes, G/L IDs, and Zone Models
Charge Selector	Charges	Value Maps and Zone Models
Charge Offer	Charge Selectors and Charges	RUMs, Deliverables, Tax Codes, Tax Suppliers, Provisioning Tags, and Extended Attribute Templates
Discount	Pricing, Discount Trigger, and Discount Filter	Balance Elements, Tax Codes, and G/L IDs
Discount Selector	Discounts	None
Discount Offer	Discount Selectors and Discounts	Provisioning Tags and Extended Attribute Templates

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 31-1 (Cont.) Prerequisite Data for Importing Pricing Components

Pricing Component	Prerequisite Pricing Component	Prerequisite Setup Component
Discount Offer Exclusion	Discount Offer	None
Bundle	Charge Offers, Discount Offers, and Subscription Terms	Extended Attribute Templates
Bundle Dependency	Bundles	None
Bundle Transition	Bundles	None
Package	Bundles and Subscription Terms	Balance Elements and Extended Attribute Templates
Package Discount Restriction	Bundles	None
Package List	Packages	None
Package Transition	Packages	None
Subscription Terms	None	G/L IDs and Balance Elements
Subscription Terms Auto Renew	Subscription Terms	None
Chargeshare	Pricing, Trigger, Filter, and Charge Selector Spec	Balance Elements, Tax Codes, and G/L IDs
Chargeshare Offer	Distribution Charge	Extended Attribute Templates

About the XSD Files

ImportExportPricing uses separate XSD files for importing or exporting pricing components, setup components, and profile data. The XSD files are available in their respective directories in the *PDC_home***/apps/xsd** directory, where *PDC_home* is the directory in which the PDC software is installed. The XSD files describe 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.

Prerequisites for Using the ImportExportPricing Utility

To use **ImportExportPricing**, you must:

- Ensure that the XML files used by this utility conform to the format detailed in the corresponding XSD files. See "About the XSD Files" for more information.
- Ensure *PDC_home***/apps/bin** is in your PATH environment variable.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Exporting Pricing and Setup Components from PDC

Exporting by using **ImportExportPricing** involves extracting the pricing and setup components from the PDC database into XML files.

Before exporting pricing or setup components, consider the following:

- You can export all or a subset of pricing and setup components.
- When exporting pricing components, the utility does not export the setup component referenced by pricing components. You must export the setup components explicitly.

ImportExportPricing enables you to export the following from a PDC database to an XML file:

- All pricing and setup components. See "Exporting All Pricing and Setup Components" for more information.
- A subset of pricing and setup components. See "Exporting a Subset of Pricing and Setup Components" for more information.

Exporting All Pricing and Setup Components

To export all pricing and setup components to XML files, enter the following command:

ImportExportPricing -export [*Prefix*] **-appsvruser** *ApplicationServerUserName* **-pdcuser** *PdcUserName*

See "ImportExportPricing" for a description of the parameters used in this command.

For example, the following command exports all pricing components, setup components, and profile data in the PDC database in a Linux environment to the **MyData_export_pricing.xml**, **MyData_export_config.xml**, and **MyData_export_profile.xml** files, respectively:

```
ImportExportPricing -export MyData -appsvruser weblogic -pdcuser pdcuser
```

Exporting a Subset of Pricing and Setup Components

You can export specific pricing and setup components based on the object type, modification time, and the user who modified the data. You can also filter the exported data based on the object name.

For sample commands for exporting a subset of data, see the following:

- Example 1: Exporting a Subset of Pricing Components
- Example 2: Exporting Only Setup Components Managed in BRM
- Example 3: Exporting Only G/L IDs
- Example 4: Exporting All Pricing Reference Objects
- Example 5: Exporting Only Metadata Objects
- Example 6: Exporting Setup Components Modified After a Specific Time
- Example 7: Exporting Pricing Components Modified by Specific Users
- Example 8: Exporting Only Extended Attribute Template Objects

Copyright © 2017, 2026, Oracle and/or its affiliates.

Example 1: Exporting a Subset of Pricing Components

The following example exports discount and charge pricing components from the PDC database to the **MyPricingComponents_export_pricing.xml** file:

```
ImportExportPricing -export MyPricingComponents -pricing
ALTERATION_RATE_PLAN,CHARGE_RATE_PLAN -appsvruser weblogic -pdcuser pdcuser
```

Example 2: Exporting Only Setup Components Managed in BRM

The following example exports all BRM-managed setup components from the PDC database into the **MySetupComponents_export_config.xml** file:

```
ImportExportPricing -export MySetupComponents -brmObject
```

Example 3: Exporting Only G/L IDs

The following example exports G/L IDs into the **GL_IDs_export_config.xml** file:

```
ImportExportPricing -export GL_IDs -brmobject GLID
```

Example 4: Exporting All Pricing Reference Objects

The following example exports the ChargeOffer-RealtimeUsage and ChargeOffer-Batch reference objects, including the corresponding metadata and setup components, from the PDC database to the **MyChargeOffers_export_pricing.xml**, **MyChargeOffers_export_metadata.xml**, and **MyChargeOffers_export_config.xml** files:

```
ImportExportPricing -export MyChargeOffers -pricing CHARGE_OFFERING -name "ChargeOffer-
RealtimeUsage,ChargeOffer-Batch" -expAllRefs
```

The utility uses a comma (**,**) as the default separator for the list of names. If an object name contains a comma, you must use the **-separator** option to specify a different separator character. This option must be placed before the **-name** parameter.

Example 4a: Exporting Objects with Commas in Their Names

The following example exports a single charge offer named **ChargeOffer,WithComma** by using a pipe (**|**) as the separator:

```
ImportExportPricing -export MyCommaOffer -pricing CHARGE_OFFERING -separator " | " -name
"ChargeOffer,WithComma" -expAllRefs
```

Example 5: Exporting Only Metadata Objects

The following example exports all event, service, account, and profile attribute specifications from the PDC database in a Linux environment to the **MySetupComponents_export_meta_config.xml** file:

```
ImportExportPricing -export MySetupComponents -metadata
```

Example 6: Exporting Setup Components Modified After a Specific Time

The following example exports all setup components modified after 2010-01-05T19:05:09GMT+05:30 from the PDC database in a Linux environment to the **MySetupComponents_export_config.xml** file:

Copyright © 2017, 2026, Oracle and/or its affiliates.

```
ImportExportPricing -export MySetupComponents -config -ma 2010-01-05T19:05:09GMT+05:30 -
appsvruser weblogic -pdcuser pdcuser
```

Example 7: Exporting Pricing Components Modified by Specific Users

The following example exports pricing components modified by User1 and User2 from the PDC database in a Linux environment to the **MyPricingComponents_export_pricing.xml** file:

```
ImportExportPricing -export MyPricingComponents -pricing -mby User1, User2 -appsvruser
weblogic -pdcuser pdcuser
```

Example 8: Exporting Only Extended Attribute Template Objects

The following example exports all ProductSpecCharacteristicTemplate objects, which define the extended attributes available to pricing components, from the PDC database in a Linux environment to the **MyExtendedAttributes_export_config.xml** file:

```
ImportExportPricing -export MyExtendedAttributes -config PSCTEMPLATE
```

Exporting Promoted and Failed Components

By default, **ImportExportPricing** exports only successfully promoted and transformed objects. Running the **ImportExportPricing** utility with the **inclFailedObjs** parameter exports the most currently promoted components in addition to the failed components.

To export only successfully promoted and transformed pricing components, run the following command:

```
ImportExportPricing  -export –pricing -inclFailedObjs -appsvruser
ApplicationServerUserName -pdcuser PdcUserName
```

Exporting Only Obsoleted Components

You can export only obsoleted versions of pricing and setup components. For example, the following command exports all obsoleted pricing components (including metadata and setup components) from PDC in a Linux environment to the **MyPricingObsoleteComponents_export_pricing.xml** file:

```
ImportExportPricing -export MyPricingObsoleteComponents -pricing -expObsolete -
appsvruser weblogic -pdcuser pdcuser
```

About Supported Pricing and Setup Components

Table 31-2 lists the type of pricing components you can import to or export from PDC using the **ImportExportPricing** utility.

Table 31-2 Supported Pricing Components and Their Representation in the Command Line and XML File

Pricing Component Type	Represented in the Command Line and XML File As ...
Charge	CHARGE_RATE_PLAN
Discount	ALTERATION_RATE_PLAN
Chargeshare	DISTRIBUTION_RATE_PLAN
Rollover	ROLLOVER_RATE_PLAN
Discount Selector	ALTERATION_RATE_PLAN_SELECTOR

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 31-2 (Cont.) Supported Pricing Components and Their Representation in the Command Line and XML File

Pricing Component Type	Represented in the Command Line and XML File As ...
Generic Selector	GENERIC_SELECTOR
Charge Selector	CHARGE_RATE_PLAN_SELECTOR
Price Selector	PRICE_MODEL_SELECTOR
APN Selector	APN_MAP
USC Selector	USC_MAP
Tax Selector	TAX_SELECTOR
Tax Exemption Selector	TAX_EXEMPTION_SELECTOR
Time Model	TIME_MODEL
Trigger	TRIGGER_SPEC
Filter	CHARGE_SELECTOR_SPEC
Discount Pricing	ALTERATION_POP_MODEL
Rollover Rules	ROLLOVER_POP_MODEL
Recurring Pricing	RECURRING_POP_MODEL
One-Time Pricing	ONE_TIME_POP_MODEL
Charge Offer	CHARGE_OFFERING
Discount Offer	ALTERATION_OFFERING
Chargeshare Offer	DISTRIBUTION_OFFERING
Discount Exclusion	ALTERATION_EXCLUSION
Subscription Terms	SUBSCRIPTION_TERMS
Subscription Terms Auto Renew	SUBSCRIPTION_TERMS_AUTO_RENEW
Bundle	BUNDLED_PRODUCT_OFFERING
Bundle Transition	BUNDLE_TRANSITION
Bundle Dependency	BUNDLE_DEPENDENCY
Package	PACKAGE_OBJ
Discount Restriction	PACKAGE_EXCLUSION
Package Transition	PACKAGE_TRANSITION
Package List	PACKAGE_LIST

Table 31-3 lists the type of setup components you can import to or export from PDC using **ImportExportPricing**.

Table 31-3 Supported Setup Components and their Representation in the Command Line and XML File

Setup Component Type	Represented in the Command Line and XML File As ...
Value Map	VALUE_MAP
Event	EVENT_ATTRIBUTE_SPEC
Service	PRODUCT_ATTRIBUTE_SPEC
Account Attribute	CUSTOMER_ATTRIBUTE_SPEC
RUM	RUM_CONFIGURATION
Impact Category for Zones	ZONE_RESULT_CONFIGURATION
Service-Event Map	ATTRIBUTE_SPEC_MAP

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 31-3 (Cont.) Supported Setup Components and their Representation in the Command Line and XML File

Setup Component Type	Represented in the Command Line and XML File As ...
Balance Element	BALANCE_ELEMENT
Deliverable	DELIVERABLE
G/L ID	GLID
Tax Code	TAX_CODE
Tax Exemption Code	TAX_EXEMPTION_CODE
Tax Supplier	TAX_SUPPLIER
Geographical Zone Model	GEO_ZONE_MODEL
Standard Zone Model	STANDARD_ZONE_MODEL
Provisioning Tag	PROVISIONING_TAG
Special Day Calendar	HOLIDAY_CALENDAR
Custom Rule	CUSTOM_ANALYZER_RULE
Profile Attribute Specification	PROFILE_ATTRIBUTE_SPEC
Extended Attribute Template	PSCTEMPLATE

Importing Pricing and Setup Components

Importing pricing and setup components using the **ImportExportPricing** utility involves retrieving data from one or more XML files and loading them into the PDC database. Your XML files must conform to the format detailed in the XSD files for pricing or setup components. See "About the XSD Files" for more information.

Before importing pricing and setup components, consider the following:

- If you import pricing and setup components together, the utility imports setup components before importing the pricing components. If setup components are not imported successfully, the utility does not import the pricing components. When importing data, you must include the **-metadata**, **-config**, **-pricing**, **-profile**, or **-customfields** parameter at the command.
- During import, the utility modifies the objects in the database:
  - If the **-ow** parameter is used and the objects already exist in the database, the utility overwrites the existing objects. The utility generates an error if the **-ow** parameter is not used and the objects already exist in the database.
  - If the object does not exist in the database, the utility creates the object, regardless of the usage of the **-ow** parameter.
  - If the **-ignoreID** parameter is used, the utility ignores the internal IDs in the XML files. The utility looks up the internal IDs by object name or generates internal IDs for new objects. Use this option to avoid conflicts when importing data from another environment.

    Use this parameter with the **-ow** parameter when importing existing data.

Copyright © 2017, 2026, Oracle and/or its affiliates.

> **Note**
>
> - Importing charge selectors by ID may result in exceptions when attempting to view them. To address this issue, use names for referencing data between objects.
> - If you encounter a **ModelData** failure while importing a Charge Selector or Generic Selector, remove all **<modelData>** entries from the XML file and re-import the data.
>
> If the PDC UI displays "*" instead of the expected value in any rules within the Charge Selector or Generic Selector, export the selector, eliminate all **<modelData>** instances from the XML file, and re-import the data.

> **Note**
>
> A successful import into PDC does not necessarily mean that the imported components were successfully published to all target engines. To check the latest status of the changeset and the status reported by each transformation engine, use **changeSetStatus** mode. See "Changeset Status Mode" for more information.

For the list and description of parameters to use with **ImportExportPricing** for importing components into PDC, see "Import Mode".

You can import pricing and setup components from a single XML file or from a directory containing multiple XML files.

Importing from a Single XML File

You can import data from a single XML file that contains your pricing and setup components.

For example, to import metadata objects from an XML file into the PDC database, run the following command:

```
ImportExportPricing -import -metadata FileName
```

where *FileName* is the file name and path to your import XML file.

The **ImportExportPricing** utility imports the data from the XML file into the PDC database and commits the data. If there is an error, all data imported before the error occurred is persisted in the database.

Importing Multiple XML Files from a Directory

The **ImportExportPricing** utility can import pricing and setup components from multiple XML files located in a directory.

> **Note**
>
> The XML files in the directory must contain only one type of configuration object: only metadata objects, only setup components, or only pricing components.

To import data from multiple XML files in a directory:

Copyright © 2017, 2026, Oracle and/or its affiliates.

1. Create your import XML files. Ensure the files contain only one type of configuration object, such as only pricing components, only setup components, or only metadata objects.
2. (Optional) Create an **import_order.cfg** file listing the order in which to import the XML files. For example, you could specify to import **chargeRatePlans.xml** before **chargeOffers.xml**.

   > **Note**
   >
   > - Ensure **import_order.cfg** does not contain empty lines.
   > - Without the file, **ImportExportPricing** imports your XML files in a random order.

3. Copy your import XML files and **import_order.cfg** file to your input directory. Your input directory can include one or more subdirectories, but the **import_order.cfg** file must be at the top level of your input directory.
4. Set the ownership and permissions of the input directory, its subdirectories, your import XML files, and **import_order.cfg** file to the PDC installation user and **chmod 755**.
5. Run the **ImportExportPricing** utility in **import** mode. For example, to import pricing setup components from multiple XML files in a directory into the PDC database:

   ```
   ImportExportPricing -import -config DirectoryName
   ```

   where *DirectoryName* is a directory present in the current location. The input path can be an absolute or relative directory location.

The **ImportExportPricing** utility imports the data from your XML files into the PDC database and commits the data. If there is an error, all data imported before the error occurred is persisted in the database.

Checking the Status of an Import Operation

Use the **ImportExportPricing** utility to check whether PDC has finished processing a changeset after you import pricing or setup components. A successful import into PDC does not necessarily mean that PDC successfully published the imported components to the target engines.

The import operation does not wait for all downstream publishing and transformation processing to finish. Use the overall changeset status and the status of each transformation engine to verify the end-to-end result.

Use **changeSetStatus** after importing pricing or setup components, especially when the import operation submits the changeset for publication. An import command can report that an XML file was imported successfully before all target engines finish processing the published changes.

To check the status of an import operation:

1. Find the changeset name or ID in the **ImportExportPricing** command output, the **ImportExportPricing** log, or the XML status report.
2. Run **ImportExportPricing** in **changeSetStatus** mode and specify the changeset name or ID.
3. Review the overall changeset status and the status of each transformation engine in the command output or XML status report.

Copyright © 2017, 2026, Oracle and/or its affiliates.

4. If the overall status is **PENDING**, or an engine status is **DISPATCHED**, run the command again later until the status changes to **SUCCEEDED** or **FAILED**.

For example, the following command checks a changeset by name:

ImportExportPricing -changeSetStatus -name changesetName

Listing Pricing or Setup Components

You can use the **ImportExportPricing** utility to display components available in the PDC database. For the list and description of parameters to use, see "Type Mode".

The following example command displays the metadata components available in the PDC database:

```
ImportExportPricing -type metadata
```

Publishing Setup Components

You can use the **ImportExportPricing** utility with the **-publish** and **-target** parameters to publish components from PDC to the specified target engines, such as the batch rating engine, real-time rating engine, or ECE. For the list and description of parameters to use for publishing components, see "Publish Mode".

The following example command publishes all setup components from PDC to ECE:

```
ImportExportPricing -publish config -target ece
```

Deleting Setup Components

You can use the **ImportExportPricing** utility with the **-delete** parameter to delete setup components from the PDC database. See "Delete Mode" for a description of parameters used in this command.

For example, the following command deletes the setup components defined in the **/Temp/ComponentsToDelete.xml** file from the PDC database:

```
ImportExportPricing -delete /Temp/ComponentsToDelete.xml
```

Deleting Pricing Profiles

You can also use the **ImportExportPricing** utility to delete pricing profiles if they are not being used by any pricing component. For example, the following command deletes the pricing profile data defined in the **/Temp/ProfilesToDelete.xml** file from the PDC database:

```
ImportExportPricing -delete /Temp/ProfilesToDelete.xml
```

Keeping Only the Latest Versions of Successfully Promoted PDC Components

You can keep only the latest versions of successfully promoted PDC components and delete all older versions in PDC or in the associated target engines and by using the **-keep** parameter. For the list and description of parameters to use for deleting older versions of PDC components, see "Keep Mode".

Copyright © 2017, 2026, Oracle and/or its affiliates.

For example, the following command keeps the last three successfully promoted versions of pricing components in the PDC database and deletes all older versions of the pricing components:

```
ImportExportPricing -keep pricing -numVersion 3
```

> **Note**
>
> PDC keeps versions of the successfully promoted components if they are available in the PDC database. For example, if you want to keep the latest two versions of successfully promoted components and there is only one version available in the PDC database, PDC keeps only that one version.

Obsoleting PDC Components

You can use the **ImportExportPricing** utility to obsolete PDC components with a Promoted status if they are not referenced by any other pricing components.

To obsolete PDC components:

1. Export the PDC component that you want to obsolete into an XML file. See "Exporting Pricing and Setup Components from PDC" for more information.
2. In the XML file, set the **<obsolete>** element for the PDC component to **true**:

   ```
   <obsolete>true</obsolete>
   ```
3. Import the PDC component into the PDC database with the **-ow** parameter. See "Importing Pricing and Setup Components" for more information.

   For example:

   ```
   ImportExportPricing -import -pricing export_pricing.xml -ow
   ```

   The imported PDC component is obsoleted.

   > **Note**
   >
   > If you want to reuse an obsoleted PDC component, set the **<obsolete>** element for that component to **false** and import it into the PDC database.

Copyright © 2017, 2026, Oracle and/or its affiliates.

32

XML Examples of Creating Pricing Components

Learn how to use the XML examples as a guideline for creating your own pricing components using the Oracle Communications Pricing Design Center (PDC) XML Pricing Interface.

Topics in this document:

- Bundle Configuration Examples
- Charge Offer Configuration Examples
- Discount Offer Configuration Examples
- Package Configuration Examples
- Selector Configuration Examples

For a list of PDC XML tags that you can use to create pricing components, see "PDC XML Tags".

Bundle Configuration Examples

To configure bundle functionality by using the PDC XML Pricing Interface, see the following examples:

- Activating Offers in Bundles
- Configuring Add-On Charge Offers In Bundles
- Configuring Proration for Offers in Bundles
- Purchasing the Same Offer in a Bundle Multiple Times
- Setting Charge Offer Cycle Alignment for Reactivated Bundles
- Transitioning Bundles

Activating Offers in Bundles

When creating bundles in XML, you can specify whether to activate all charge offers in a bundle when one of the charge offers is activated on first usage. To do so, you use the **<firstUsageActivation>** element under the **<bundledProductOffering>** element.

Set the **<firstUsageActivation>** element to one of the following:

- **true**: Activates all charge offers in the bundle upon first usage of any charge offer in the bundle.

  For example, assume a bundle contains charge offers A, B, and C. If the first service a customer uses is in charge offer B, all charge offers (A, B, and C) are activated at that time.

- **false**: When one charge offer in a bundle is activated on first usage, the other charge offers in the bundle are not activated.

Copyright © 2017, 2026, Oracle and/or its affiliates.

For example, assume a bundle contains charge offers A, B, and C. If the first service a customer uses is in charge offer B, only charge offer B is activated at that time. This is the default.

The following shows an example of a bundle with the **<firstUsageActivation>** element set:

```
<bundledProductOffering xmlns:pdc="http://xmlns.oracle.com/communications/
platform/model/pricing">
   <name>Bundle for Data</name>
   <internalId>InternalID</internalId>
   <pricingProfileName>Product Offering</pricingProfileName>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <timeRange>0/inf</timeRange>
   <productSpecName>TelcoGsmTelephony</productSpecName>
   <billOnPurchase>false</billOnPurchase>
   <customize>OPTIONAL</customize>
   <firstUsageActivation>true</firstUsageActivation>
   <groupBalanceElements>false</groupBalanceElements>
   <bundledProductOfferingItem>
      .....
      .....
      .....
   </bundledProductOfferingItem>
</bundledProductOffering>
```

Configuring Add-On Charge Offers In Bundles

When creating bundles in XML, you can include add-on charge offers. By default, all charge offers in bundles are base charge offers, which means they can be purchased without any prerequisites. Add-on charge offers require customers to own a valid base charge offer.

You configure an add-on charge offer in a bundle by doing the following:

1. Including the **<addOnOfferingValidityMode>** element under **<bundledProductOfferingItem>**.
2. Specifying the add-on charge offer's validity. The add-on charge offer's validity start date is the end date of a charge offer that you specify. For example, assume charge offer A has a validity period from June 1 through June 15. If you specify to align add-on charge offer B's validity period with charge offer A, charge offer B's validity start date would be June 15.

   You specify the charge offer on which to align an add-on charge offer's validity dates by setting the **<addOnOfferingValidityMode>** element to one of the following:

   - **BASE_OFFER**: Aligns the validity dates with the specified base charge offer.
   - **BASE_OFFER_EXPIRY_FIRST**: Aligns the validity dates with the active base charge offer that expires first.
   - **BASE_OFFER_EXPIRY_LAST**: Aligns the validity dates with the active base charge offer that expires last.
   - **BASE_OR_ADD_ON_OFFER_EXPIRY_FIRST**: Aligns the validity dates with the active charge offer that expires first.
   - **BASE_OR_ADD_ON_OFFER_EXPIRY_LAST**: Aligns the validity dates with the active charge offer that expires last.

Copyright © 2017, 2026, Oracle and/or its affiliates.

3. If you set **<addOnOfferingValidityMode>** to **BASE_OFFER**, add the **<baseChargeOfferingName>** element set to the name of the base charge offer on which to set validity dates.

> **Note**
>
> Do not include the **<baseChargeOfferingName>** element for the other **<addOnOfferingValidityMode>** values: **BASE_OFFER_EXPIRY_FIRST**, **BASE_OFFER_EXPIRY_LAST**, **BASE_OR_ADD_ON_OFFER_EXPIRY_FIRST**, and **BASE_OR_ADD_ON_OFFER_EXPIRY_LAST**.

4. Setting the validity period for all charges in the add-on charge offer to the default value (immediate to never expires).

   To do so, add the following elements under **<purchaseStart>**, **<purchaseEnd>**, **<usageStart>**, **<usageEnd>**, **<cycleStart>**, and **<cycleEnd>**:

   - **<offset>** set to **0**
   - **<mode>** set to **NOW_TO_NEVER**

The following shows sample XML for creating an add-on charge offer that uses the same validity period as the **Sample01** base charge offer:

```
<bundledProductOfferingItem>
   <purchaseStart>
      <offset>0</offset>
      <mode>NOW_TO_NEVER</mode>
   </purchaseStart>
   <purchaseEnd>
      <offset>0</offset>
      <mode>NOW_TO_NEVER</mode>
   </purchaseEnd>
   <usageStart>
      <offset>0</offset>
      <mode>NOW_TO_NEVER</mode>
   </usageStart>
   <usageEnd>
      <offset>0</offset>
      <mode>NOW_TO_NEVER</mode>
   </usageEnd>
   <cycleStart>
      <offset>0</offset>
      <mode>NOW_TO_NEVER</mode>
   </cycleStart>
   <cycleEnd>
      <offset>0</offset>
      <mode>NOW_TO_NEVER</mode>
   </cycleEnd>
   <status>1</status>
   <statusCode>0</statusCode>
   <renewalMode>false</renewalMode>
   <quantity>1.0</quantity>
   <purchaseChargeAdjustment>0.0</purchaseChargeAdjustment>
   <usageChargeAdjustment>0.0</usageChargeAdjustment>
   <cycleChargeAdjustment>0.0</cycleChargeAdjustment>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
   <purchaseMode>NEW_INSTANCE</purchaseMode>
   <addOnOfferingValidityMode>BASE_OFFER</addOnOfferingValidityMode>
   <baseChargeOfferingName>Sample01</baseChargeOfferingName>
   <chargeOfferingName>MyChargeOffer</chargeOfferingName>
</bundledProductOfferingItem>
```

Configuring Proration for Offers in Bundles

When creating bundles in XML, you can set how to prorate cycle charges and discounts at the offer level.

To configure proration for an offer in a bundle, set the **<prorationBasedOn>** element under **<bundledProductOfferingItem>** to one of the following:

- **PRORATE_30_DAYS**: Prorated charges and discounts are based on a 30-day month.
- **PRORATE_DAYS_IN_MONTH**: Prorated charges and discounts are based on the actual number of days in a month, such as 30 for June or 31 for May.
- Not Set: Prorated charges are calculated according to the system-wide setting in the Connection Manager (CM) **pin.conf** file. See "Examples of 30-Day-Based Proration" in *BRM Configuring and Running Billing*.

The following shows sample XML for configuring the **Sample01** charge offer to use 30-day proration:

```
<bundledProductOfferingItem>
   ...
   <status>1</status>
   <statusCode>0</statusCode>
   <renewalMode>false</renewalMode>
   <quantity>1.0</quantity>
   <purchaseChargeAdjustment>0.0</purchaseChargeAdjustment>
   <usageChargeAdjustment>0.0</usageChargeAdjustment>
   <cycleChargeAdjustment>0.0</cycleChargeAdjustment>
   <prorationBasedOn>PRORATE_30_DAYS</prorationBasedOn>
   <purchaseMode>NEW_INSTANCE</purchaseMode>
   <addOnOfferingValidityMode>BASE_OFFER_EXPIRY_FIRST</
addOnOfferingValidityMode>
   <chargeOfferingName>Sample01</chargeOfferingName>
</bundledProductOfferingItem>
```

Purchasing the Same Offer in a Bundle Multiple Times

When creating bundles in XML, you can specify what happens if customers purchase the same charge offer or discount offer more than once.

For both charge offers and discount offers, you can specify whether the additional offer is purchased as a new subscription, a replacement of the existing subscription, or an extension to the existing subscription. To do so, in the definition of the bundle containing the charge offer or discount offer, set the value for the **<purchaseMode>** element under the **<bundledProductOfferingItem>** element.

Copyright © 2017, 2026, Oracle and/or its affiliates.

> **⚠ Caution**
>
> The following values can be applied to one-time charges only: **OVERWRITE**, **LONGEST_DATE**, **LONGEST_DATE_NEW_SUB_BAL**, **EXTEND**, and **EXTEND_NEW_SUB_BAL**. These values cannot be applied to cycle or usage charges.

To create a new subscription or replace an existing one, set **<purchaseMode>** to one of these values:

- **NEW_INSTANCE**: The additional charge offer or discount offer is purchased as a new, unrelated subscription. The balance impacts and validity periods of the old and new subscriptions are entirely independent. This option is the default.
- **OVERWRITE**: The additional charge offer or discount offer replaces the existing subscription. The existing subscription is canceled, any configured proration is applied, and the new subscription is created, with all balance impacts and validity periods set as though for a new purchase.

  For example, on June 1, a customer purchases an add-on bundle that grants 3 GB of data and is valid for seven days. On June 3, after using 1 GB, they repurchase the bundle. The old charge offer is canceled, any remaining balance is prorated, and the new 3 GB balance is created, with a validity end date of June 10.

To extend the old subscription when the additional charge offer is purchased within the specified grace period, set **<purchaseMode>** to one of these values:

- **LONGEST_DATE**: The additional charge offer is purchased as an extension to the existing subscription if it is purchased within the grace period. The new balance impacts are added to the existing balance group. The balance's validity period is then set to either the old validity end date or the new validity end date, whichever is later.

  You specify the grace period under the **<bundledProductOfferingItem>** by setting the **<gracePeriodOffset>** element to the amount of time and the **<gracePeriodUnit>** element to **Seconds**, **Minutes**, **Hours**, or **Days**.

  For example, on June 1, a customer purchases an add-on bundle that grants 3 GB of data, is valid for seven days, and has a grace period of four days. On June 3, after using 1 GB, they purchase the bundle again. The new 3 GB balance is added to the remaining balance, for a total of 5 GB. BRM compares the validity periods of the old and new charge offers and sets the validity end date for the total balance to the later of the two end dates, June 10.
- **LONGEST_DATE_NEW_SUB_BAL**: The additional charge offer is purchased as an extension to the existing subscription if it is purchased within the grace period. The new balance impacts are added to a new balance group. The validity period for the new balance starts on the purchase date and ends on either the old or new validity end date, whichever is later.

  You specify the grace period under the **<bundledProductOfferingItem>** by setting the **<gracePeriodOffset>** element to the amount of time and the **<gracePeriodUnit>** element to **Seconds**, **Minutes**, **Hours**, or **Days**.

  For example, on June 1, a customer purchases an add-on bundle that grants 3 GB of data, is valid for seven days, and has a grace period of four days. On June 3, after using 1 GB, they purchase the bundle again. The new 3 GB balance is added to a new balance group, separate from the remaining 2 GB in the old balance group. The new balance is valid starting on June 1. BRM compares the validity periods of the old and new charge offers

Copyright © 2017, 2026, Oracle and/or its affiliates.

and sets the validity end date for the new balance to the later of the two end dates, June 10. The old balance retains the original validity end date.

- **EXTEND**: The additional charge offer is purchased as an extension to the existing subscription if it is purchased within the grace period. The new balance impacts are added to the existing balance group. The balance's validity period is then set by adding the new validity period to whatever remains of the old validity period.

  You specify the grace period under the **<bundledProductOfferingItem>** by setting the **<gracePeriodOffset>** element to the amount of time and the **<gracePeriodUnit>** element to **Seconds**, **Minutes**, **Hours**, or **Days**.

  For example, on June 1, a customer purchases an add-on bundle that grants 3 GB of data, is valid for seven days, and has a grace period of four days. On June 3, after using 1 GB, they purchase the bundle again. The new 3 GB balance is added to the remaining balance, for a total of 5 GB. BRM adds the validity period of the new charge offer to the old charge offer and sets the validity end date for the total balance to June 15.

- **EXTEND_NEW_SUB_BAL**: The additional charge offer is purchased as an extension to the existing subscription if it is purchased within the grace period. The new balance impacts are added to a new balance group. The validity period for the new balance starts on the end date for the old balance's validity, and the new end date is set by adding the new validity period to whatever remains of the old validity period.

  You specify the grace period under the **<bundledProductOfferingItem>** by setting the **<gracePeriodOffset>** element to the amount of time and the **<gracePeriodUnit>** element to **Seconds**, **Minutes**, **Hours**, or **Days**.

  For example, on June 1, a customer purchases an add-on bundle that grants 3 GB of data, is valid for seven days, and has a grace period of four days. On June 3, after using 1 GB, they purchase the bundle again. The new 3 GB balance is added to a new balance group, separate from the remaining 2 GB in the old balance group. The new balance is valid starting June 8. BRM adds the validity period of the new charge offer to the old charge offer and sets the validity end date for the new balance to June 15. The old balance retains the original validity end date.

The following shows a sample of the XML for a charge offer in a bundle with **<purchaseMode>** set to **EXTEND**, **<gracePeriodOffset>** set to **5**, and **<gracePeriodUnit>** set to **Day**.

```
<BundledProductOffering>
   <name>Extend Bundle</name>
   <description>Bundle with Extend Purchase Mode</description>
   <internalId>InternalID</internalId>
   <obsolete>false</obsolete>
   <timeRange>0/inf</timeRange>
   <productSpecName>TelcoGsm</productSpecName>
   <billOnPurchase>false</billOnPurchase>
   <customize>OPTIONAL</customize>
   <groupBalanceElements>false</groupBalanceElements>
   <bundledProductOfferingItem>
      <purchaseStart>
         <offset>0</offset>
         <mode>NOW_TO_NEVER</mode>
      </purchaseStart>
      <purchaseEnd>
         <offset>0</offset>
         <mode>NOW_TO_NEVER</mode>
      </purchaseEnd>
      <usageStart>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
            <offset>0</offset>
            <mode>NOW_TO_NEVER</mode>
         </usageStart>
         <usageEnd>
            <offset>0</offset>
            <mode>NOW_TO_NEVER</mode>
         </usageEnd>
         <cycleStart>
            <offset>0</offset>
            <mode>NOW_TO_NEVER</mode>
         </cycleStart>
         <cycleEnd>
            <offset>0</offset>
            <mode>NOW_TO_NEVER</mode>
         </cycleEnd>
         <status>1</status>
         <statusCode>0</statusCode>
         <renewalMode>true</renewalMode>
         <quantity>1.0</quantity>
         <purchaseChargeAdjustment>0.0</purchaseChargeAdjustment>
         <usageChargeAdjustment>0.0</usageChargeAdjustment>
         <cycleChargeAdjustment>0.0</cycleChargeAdjustment>
         <purchaseMode>EXTEND</purchaseMode>
         <gracePeriodOffset>5</gracePeriodOffset>
         <gracePeriodUnit>Day</gracePeriodUnit>
         <chargeOfferingName>Charge Offer - Purchase Fee</chargeOfferingName>
      </bundledProductOfferingItem>
</BundledProductOffering>
```

The following shows a sample of the XML for a discount offer in a bundle with **<purchaseMode>** set to **NEW_INSTANCE**.

```
<bundledProductOfferingItem>
    <purchaseStart>
        <offset>0</offset>
        <mode>NOW_TO_NEVER</mode>
    </purchaseStart>
    <purchaseEnd>
        <offset>0</offset>
        <mode>NOW_TO_NEVER</mode>
    </purchaseEnd>
    <usageStart>
        <offset>0</offset>
        <mode>NOW_TO_NEVER</mode>
    </usageStart>
    <usageEnd>
        <offset>0</offset>
        <mode>NOW_TO_NEVER</mode>
    </usageEnd>
    <cycleStart>
        <offset>0</offset>
        <mode>NOW_TO_NEVER</mode>
    </cycleStart>
    <cycleEnd>
        <offset>0</offset>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
        <mode>NOW_TO_NEVER</mode>
    </cycleEnd>
    <status>1</status>
    <statusCode>0</statusCode>
       <renewalMode>true</renewalMode>
    <quantity>1.0</quantity>
    <purchaseChargeAdjustment>0.0</purchaseChargeAdjustment>
    <usageChargeAdjustment>0.0</usageChargeAdjustment>
    <cycleChargeAdjustment>0.0</cycleChargeAdjustment>
    <purchaseMode>NEW_INSTANCE</purchaseMode>
    <alterationOfferingName>Discount Offer - Usage Discount</
alterationOfferingName>
</bundledProductOfferingItem>
```

Setting Charge Offer Cycle Alignment for Reactivated Bundles

When creating bundles in XML, you can specify when the cycle should align for charge offers if a customer suspends then reactivates the bundle.

In the definition of the bundle containing the charge offer, set the value for the **<renewalMode>** element under the **<bundledProductOfferingItem>** element to **true** or **false**. This setting is not valid for discount offers.

If **<renewalMode>** is set to **false**, the cycle aligns as described in "Aligning Recurring Charges and Product Validity to a Specific Day of the Month".

If **<renewalMode>** is set to **true**, the following scenarios are possible:

- When **<cycleFeeFlag>** under **chargeRatePlan** is set to **1**:
  - If **<purchaseCycleDOM>** under **chargeOffering** is not set, the cycle aligns with the reactivation date.
  - If **<purchaseCycleDOM>** is set, the cycle aligns with either **<purchaseCycleDOM>** or the reactivation date, whichever is later.
- When **<cycleFeeFlag>** is set to **0**, the cycle aligns with the billing date.

For a sample of the XML for a charge offer in a bundle with **<renewalMode>** set to **true**, see "Purchasing the Same Offer in a Bundle Multiple Times".

Transitioning Bundles

When creating bundle transitions in XML, you can configure how PDC applies charges when customers transition bundles in the middle of their billing cycle by using the **<prorationType>** element under the **<bundleTransition>** element.

Set the **<prorationType>** element to one of the following:

- **PRORATE_CHARGE**: Specifies to prorate charges for both bundles. This is the default.
- **ORIGINAL_CHARGE**: Specifies to apply the full charges from the original bundle.
- **TRANSFER_CHARGE**: Specifies to apply the full charges from the new bundle.

The following shows an example of a bundle transition with the **<prorationType>** element set:

```
<bundleTransition xmlns:pdc="http://xmlns.oracle.com/communications/platform/
model/pricing">
   <name>sample package transition</name>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
   <internalId>InternalID</internalId>
   <pricingProfileName>Subscription</pricingProfileName>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <active>true</active>
   <timeRange>0/inf</timeRange>
   <waivePurchaseFee>true</waivePurchaseFee>
   <waiveCancelFee>true</waiveCancelFee>
   <productSpecName>TelcoGsmTelephony</productSpecName>
   <prorationType>ORIGINAL_CHARGE</prorationType>
   <type>GENERATION_CHANGE</type>
   <fromBundleName>Bundle A</fromPackageName>
   <toBundleName>Bundle B</toPackageName>
</bundleTransition>
```

Charge Offer Configuration Examples

To configure charge offer functionality by using the PDC XML Pricing Interface, see the following examples:

- Aligning Recurring Charges and Product Validity to a Specific Day of the Month
- Enforcing or Exceeding Credit Limits
- Offering Hourly Balance Impacts
- Consuming Granted Allowances Before Applying Usage Charges
- Prorating Recurring Fees and Rollovers
- Setting Full Day Proration
- Setting Taxation for Charges
- Using a Tax Exemption Selector
- Sharing Products Automatically with Sharing Group Members
- Using Date Ranges for Versioning
- Applying Price Tags to Charges
- Specifying Validity Period to Start from Event Occurrence Date
- Configuring the Automatic Renewal of Allowances

Aligning Recurring Charges and Product Validity to a Specific Day of the Month

When creating charge offers in XML, you can use the **<purchaseCycleDOM>** and **<cycleFeeFlag>** 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 offer validity to a specific day of the month, set **<purchaseCycleDOM>** to an integer from 1-31 and set **<cycleFeeFlag>** to **1**.
- To align recurring charges and offer validity to the billing date, set **<purchaseCycleDOM>** to **0**. The value of **<cycleFeeFlag>** is irrelevant in this scenario.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- To align recurring charges to the billing date but set offer validity to a specific day of the month, set **<purchaseCycleDOM>** to an integer from 1-31 and set **<cycleFeeFlag>** to **0**.

> **⚠ Caution**
>
> You cannot change the **<purchaseCycleDOM>** value for a charge offer after it has been purchased. If you create a new version of the charge offer with a different value in **<purchaseCycleDOM>**, 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 apply recurring charges for a charge offer on the 22nd day of the month, the XML input file would include the following:

```
<chargeRatePlan xmlns:pdc="http://xmlns.oracle.com/communications/platform/
model/pricing">
    <name>Charge for a Specific Day of the Month</name>
    <pricingProfileName>Subscription</pricingProfileName>
    <priceListName>Default</priceListName>
    <obsolete>false</obsolete>
    <applicableRums>Occurrence</applicableRums>
    <applicableQuantity>ORIGINAL</applicableQuantity>
    <taxTime>NONE</taxTime>
    <todMode>START_TIME</todMode>
    <applicableQtyTreatment>CONTINUOUS</applicableQtyTreatment>
    <permittedName>TelcoGsmTelephony</permittedName>
    <permittedType>PRODUCT</permittedType>
    <eventName>EventBillingProductFeeCycleCycle_forward_monthly</eventName>
    <cycleFeeFlag>1</cycleFeeFlag>
    <billOffset>0</billOffset>
    <subscriberCurrency>

        ...

    </subscriberCurrency>
</chargeRatePlan>
<chargeOffering xmlns:pdc="http://xmlns.oracle.com/communications/platform/
model/pricing">
    <name>Charge Offer for a Specific Day of the Month</name>
    <pricingProfileName>Product Offering</pricingProfileName>
    <priceListName>Default</priceListName>
    <obsolete>false</obsolete>
    <timeRange>0/inf</timeRange>
    <productSpecName>TelcoGsmTelephony</productSpecName>
    <offerType>SUBSCRIPTION</offerType>
    <priority>1</priority>
    <partial>false</partial>
    <purchaseMin>-1.0</purchaseMin>
    <purchaseMax>-1.0</purchaseMax>
    <ownMin>-1.0</ownMin>
    <ownMax>-1.0</ownMax>
    <applicableQuantity>REMAINING</applicableQuantity>
    <purchaseCycleDOM>22</purchaseCycleDOM>
    <dateRangeImpactType>INSTANTIATED_DATE</dateRangeImpactType>
    <chargeEventMap>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
        <eventName>EventBillingProductFeeCycleCycle_forward_monthly</
eventName>
        <validIfCancelled>false</validIfCancelled>
        <validIfInactive>false</validIfInactive>
        <timezoneMode>EVENT</timezoneMode>
        <minQuantity>0</minQuantity>
        <minQuantityUnit>NONE</minQuantityUnit>
        <incrementQuantity>1</incrementQuantity>
        <incrementQuantityUnit>NONE</incrementQuantityUnit>
        <roundingMode>NEAREST</roundingMode>
        <prorateFirst>PRORATE_CHARGE</prorateFirst>
        <prorateLast>PRORATE_CHARGE</prorateLast>
        <chargeRatePlanInfo>
            <targetEngine>RRE</targetEngine>
        </chargeRatePlanInfo>
        <chargeRatePlanName>Charge for a Specific Day of the Month</
chargeRatePlanName>
    </chargeEventMap>
</chargeOffering>
</pdc:PricingObjectsJXB>
```

If a customer with a billing date of the 14th of the month purchased the charge offer in this example on January 18, the first cycle forward fee would be applied on January 18, prorated to cover the period between January 18 and January 22 (the **<purchaseCycleDOM>** 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 **<cycleFeeFlag>** 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 charge offer's duration, it would only be valid until the 22nd of the month (the **<purchaseCycleDOM>** date.) If the charge offer was valid for three months, the final fee would be prorated to cover the period from March 14 to March 22.

> **Note**
>
> For charge offers in bundles 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 **<renewalMode>** element under the **<bundledProductOfferingItem>** element. See "Setting Charge Offer Cycle Alignment for Reactivated Bundles" for information about how **<renewalMode>** interacts with **<purchaseCycleDOM>** and **<cycleFeeFlag>**.

Enforcing or Exceeding Credit Limits

When creating charges in XML, you can specify whether to enforce credit limits, or let customers exceed them.

To do so, set the **<enforceCreditLimit>** element under the price tier to one of the following:

- **NORMAL**: Enforce the credit limit. The subscription succeeds, but customers do not exceed their credit limit. Prorate the subscription according to the available balance.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **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 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 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 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 next cycle.
- **INSUFFICIENT_BALANCE**: 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 minimum amount for currency resources in the **<creditLimitMinimumAmount>** element. By default, this is set to **0**, which means the subscription fails if the balance is less than the full amount. **<creditLimitMinimumAmount>** is not applicable for noncurrency resources.
  - Whether or not to prorate the service according to the minimum amount in the **<creditLimitProrateQuantity>** element. By default, this is set to **true**, and the service will be prorated. Set this to **false** to grant the full service for the minimum amount.

  This option is useful 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 stopgap until they have enough balance to afford the full service.

For the RENTAL_FAILURE, AUTO_RENEW_CANCEL, and INSUFFICIENT_BALANCE options, you can also configure the following:

- The maximum number of times to retry the subscription before it fails. Configure this in the **<retryMaxCount>** element of the charge offer. The default is **0**, which sends the failure notification immediately without retrying.
- The frequency of subsequent retries. Configure this in the **<retryOffset>** and **<retryUnit>** elements of the charge offer.

Copyright © 2017, 2026, Oracle and/or its affiliates.

The default values are **1** for **<retryOffset>** and **DAY** for **<retryUnit>**. The possible values for **<retryUnit>** are:

- SECOND
- MINUTE
- HOUR
- DAY
- WEEK
- MONTH
- YEAR

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 **MaxRetryCount** 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" for more information about setting this parameter.

Table 32-1 shows the mapping between **<enforceCreditLimit>** element values and the corresponding UI labels.

Table 32-1 Mapping XML Values and UI Labels

XML Value	UI Label
NORMAL	Prorate service according to available balance
DEFAULT	Use available balance, track remainder as negative account balance
LOAN	Use available balance, grant loan for remainder
DEDUCT_RENTAL	Use available balance, track remainder as outstanding
OUTSTANDING_AMOUNT	Do not use available balance, track entire amount as outstanding
RENTAL_FAILURE	Subscription fails
AUTO_RENEW_CANCEL	Subscription fails and is canceled
SKIP_CYCLE	Skip subscription charges until next cycle
INSUFFICIENT_BALANCE	Charge minimum configured amount

The following XML example specifies to charge 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.

```
<chargeRatePlan xmlns:pdc="http://xmlns.oracle.com/communications/platform/
model/pricing">
   <name>Credit Limit Sample Rate Plan</name>
   <internalId>InternalID</internalId>
   <pricingProfileName>Subscription</pricingProfileName>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <applicableRums>RECURRING</applicableRums>
   <applicableQuantity>ORIGINAL</applicableQuantity>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
   <taxTime>NONE</taxTime>
   <todMode>START_TIME</todMode>
   <applicableQtyTreatment>CONTINUOUS</applicableQtyTreatment>
   <permittedName>TelcoGsmTelephony</permittedName>
   <permittedType>PRODUCT</permittedType>
   <eventName>EventBillingProductFeeCycleCycle_forward_monthly</eventName>
   <cycleFeeFlag>0</cycleFeeFlag>
   <billOffset>0</billOffset>
   <subscriberCurrency>
      <currencyCode>USD</currencyCode>
      <crpRelDateRange>
         <absoluteDateRange>
            <startDate>0</startDate>
            <endDate>inf</endDate>
         </absoluteDateRange>
         <crpCompositePopModel>
            <name>Pricing</name>
            <recurringPopModel>
               <priceTier>
                  <lowerBound>NO_MIN</lowerBound>
                  <tierBasis>
                     <rumTierExpression/>
                  </tierBasis>
                  <rumName>RECURRING</rumName>
                  <enforceCreditLimit>INSUFFICIENT_BALANCE</
enforceCreditLimit>
                  <creditLimitProrateQuantity>true</
creditLimitProrateQuantity>
                  <tierRange>
                     <upperBound>NO_MAX</upperBound>
                     <recurringCharge>
                        <price>20.0</price>
                        <unitOfMeasure>NONE</unitOfMeasure>
                        <balanceElementNumCode>840</balanceElementNumCode>
                        <discountable>true</discountable>
                        <priceType>CONSUMPTION</priceType>
                        <proratable>true</proratable>
                        <impactType>SCALED</impactType>
                        <creditLimitMinimumAmount>10</
creditLimitMinimumAmount>
                     </recurringCharge>
                     <recurringCharge>
                        <price>-30.0</price>
                        <unitOfMeasure>NONE</unitOfMeasure>
                        <balanceElementNumCode>840</balanceElementNumCode>
                        <discountable>true</discountable>
                        <priceType>GRANT</priceType>
                        <priceValidity>
                           <startValidityMode>IMMEDIATE</startValidityMode>
                           <endValidityMode>NEVER</endValidityMode>
                           <validityRange>0/inf</validityRange>
                           <relativeStartOffset>-1</relativeStartOffset>
                           <relativeEndOffset>-1</relativeEndOffset>
                        </priceValidity>
                        <proratable>true</proratable>
                        <impactType>SCALED</impactType>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
                        <creditLimitMinimumAmount>-40</
creditLimitMinimumAmount>
                     </recurringCharge>
                  </tierRange>
               </priceTier>
            </recurringPopModel>
         </crpCompositePopModel>
      </crpRelDateRange>
   </subscriberCurrency>
</chargeRatePlan>
<chargeOffering xmlns:pdc="http://xmlns.oracle.com/communications/platform/
model/pricing">
   <name>Credit Limit Sample Offer</name>
   <internalId>InternalID</internalId>
   <pricingProfileName>Product Offering</pricingProfileName>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <timeRange>0/inf</timeRange>
   <productSpecName>TelcoGsmTelephony</productSpecName>
   <offerType>SUBSCRIPTION</offerType>
   <priority>0</priority>
   <partial>false</partial>
   <purchaseMin>-1.0</purchaseMin>
   <purchaseMax>-1.0</purchaseMax>
   <ownMin>-1.0</ownMin>
   <ownMax>-1.0</ownMax>
   <applicableQuantity>REMAINING</applicableQuantity>
   <purchaseCycleDOM>0</purchaseCycleDOM>
   <expiryNotification>true</expiryNotification>
   <subscriptionDueNotification>true</subscriptionDueNotification>
   <dateRangeImpactType>EVENT_DATE</dateRangeImpactType>
   <retryMaxCount>3</retryMaxCount>
   <retryUnit>HOUR</retryUnit>
   <retryOffset>6</retryOffset>
   <chargeEventMap>
      <eventName>EventBillingProductFeeCycleCycle_forward_monthly</eventName>
      <validIfCancelled>false</validIfCancelled>
      <validIfInactive>false</validIfInactive>
      <validIfSuspendedActive>false</validIfSuspendedActive>
      <timezoneMode>EVENT</timezoneMode>
      <minQuantity>0</minQuantity>
      <minQuantityUnit>NONE</minQuantityUnit>
      <incrementQuantity>1</incrementQuantity>
      <incrementQuantityUnit>NONE</incrementQuantityUnit>
      <roundingMode>NEAREST</roundingMode>
      <prorateFirst>PRORATE_CHARGE</prorateFirst>
      <prorateLast>PRORATE_CHARGE</prorateLast>
      <prorateCycle>PRORATE_CHARGE</prorateCycle>
      <chargeRatePlanInfo>
         <targetEngine>RRE</targetEngine>
      </chargeRatePlanInfo>
      <chargeRatePlanName>Credit Limit Sample Rate Plan</chargeRatePlanName>
      <ratePlanIID>4d0f0720</ratePlanIID>
   </chargeEventMap>
</chargeOffering>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

Offering Hourly Balance Impacts

When creating charges in XML, you can configure hourly validity periods for the balance impacts that make up charges. This is useful for creating short-lived rewards, like free games or minutes. Hourly validity applies to conditional or non-conditional noncurrency credit balance impacts.

To configure hourly validity, use the following under the **priceTierValidityPeriod** element:

- For credit balance impacts, under the **priceValidity** element, set:
  - **startValidityMode** to **PERIOD_START**
  - **endValidityMode** to **RELATIVE_TO_START**
  - **relativeEndOffsetUnit** to **HOUR**
- For conditional balance impacts, under **periodConditionExpression**, set:
  - **unit** to **HOUR**
  - **periodStartTime** to **IMMEDIATE**

The following shows sample XML that creates two conditional balance impacts defined in the price tier. The first debits USD 2.99 every day and the second credits ten games, valid for one day, once every four hours.

```
...
<priceTierValidityPeriod>
    <lowerBound>0</lowerBound>
    <validFrom>0</validFrom>
    <priceTierRange>
        <upperBound>NO_MAX</upperBound>
        <fixedCharge>
            <price>2.99</price>
            <unitOfMeasure>NONE</unitOfMeasure>
            <balanceElementNumCode>840</balanceElementNumCode>
            <discountable>true</discountable>
            <priceType>CONSUMPTION</priceType>
            <periodConditionExpression>
                <biConditionOperator>EQUAL_TO</biConditionOperator>
                <unit>DAY</unit>
                <value>1</value>
                         <periodStartTime>IMMEDIATE</periodStartTime>
            </periodConditionExpression>
        </fixedCharge>
                <fixedCharge>
                     <price>-10.0</price>
               <unitOfMeasure>NONE</unitOfMeasure>
                        <balanceElementNumCode>1000004</balanceElementNumCode>
                        <discountable>true</discountable>
                        <priceType>GRANT</priceType>
                        <priceValidity>
                             <startValidityMode>PERIOD_START</
startValidityMode>
                                <endValidityMode>RELATIVE_TO_PERIOD_START</
endValidityMode>
                                <validityRange>0/inf</validityRange>
                                <relativeStartOffset>-1</relativeStartOffset>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
                                <relativeEndOffset>1</relativeEndOffset>
                                <relativeEndOffsetUnit>DAY</
relativeEndOffsetUnit>
                        </priceValidity>
                        <periodConditionExpression>
                            <biConditionOperator>EQUAL_TO</
biConditionOperator>
                                <unit>HOUR</unit>
                                <value>4</value>
                                <periodStartTime>IMMEDIATE</periodStartTime>
                        </periodConditionExpression>
                </fixedCharge>
    </priceTierRange>
</priceTierValidityPeriod>
...
```

Because rounding time stamps for hourly charges would result in all hourly charges being valid for 24 hours, BRM does not round time stamps for charges with hourly validity, even if time stamp rounding is enabled. For example, if a customer purchases a bundle that is valid for three hours at 3pm, it is valid until 6pm. However, if BRM applied time stamp rounding, it would be valid until midnight.

Consuming Granted Allowances Before Applying Usage Charges

When configuring charges in XML, you can specify to consume a customer's granted allowance before charging any usage fees. For example, you could create a charge that first consumes a customer's monthly allowance of free movie rentals, charges $5 for the next five movie rentals, and then charges $4 for any subsequent movie rentals. You implement this functionality by adding to the charge a pricing tier with multiple ranges. For the example above, you would add a pricing tier with the following ranges:

1. Consume the entire monthly allowance of free movie rentals
2. One through five movie rentals: $5 per movie rental
3. Six movie rentals and beyond: $4 per movie rental

> **Note**
>
> Consuming granted allowances before charging for usage fees is supported by the ECE rating engine only.

To configure charges that consume granted allowances before charging usage fees, add multiple price tier ranges. Under each **<priceTierRange>** element, include an **<upperBoundExpression>** element with one of the following sets of fields:

- To consume a granted allowance, include the following entries under **<upperBoundExpression>**. Set **<balanceElementNumCode>** to the ID of the noncurrency balance element to consume, such as the ID for free movie rentals.

  ```
  <balanceTBExpression>
     <balanceElementNumCode>100002</balanceElementNumCode>
  </balanceTBExpression>
  ```

- To charge for usage, include the following entries under **<upperBoundExpression>**. Set **<value>** to the end value for the quantity range.

Copyright © 2017, 2026, Oracle and/or its affiliates.

```
    <numberTBExpression>
       <value>5</value>
    </numberTBExpression>
```

The following shows sample XML that creates a charge with two pricing tier ranges: one range for consuming a noncurrency balance element, and a second range from 0 through 600 for charging 50 cents per minute:

```
<priceTierValidityPeriod>
   <lowerBound>0</lowerBound>
   <validFrom>0</validFrom>
   <priceTierRange>
      <upperBoundExpression>
         <balanceTBExpression>
            <balanceElementNumCode>100002</balanceElementNumCode>
         </balanceTBExpression>
      </upperBoundExpression>
      <scaledCharge>
         <price>1.0</price>
         <unitOfMeasure>NONE</unitOfMeasure>
         <balanceElementNumCode>100002</balanceElementNumCode>
         <discountable>true</discountable>
         <priceType>CONSUMPTION</priceType>
         <incrementStep>1.0</incrementStep>
         <incrementRounding>NONE</incrementRounding>
      </scaledCharge>
   </priceTierRange>
   <priceTierRange>
      <upperBoundExpression>
         <numberTBExpression>
            <value>600</value>
         </numberTBExpression>
      </upperBoundExpression>
      <scaledCharge>
         <price>0.5</price>
         <unitOfMeasure>MINUTE</unitOfMeasure>
         <balanceElementNumCode>840</balanceElementNumCode>
         <discountable>true</discountable>
         <priceType>CONSUMPTION</priceType>
         <incrementStep>1.0</incrementStep>
         <incrementRounding>NONE</incrementRounding>
      </scaledCharge>
   </priceTierRange>
</priceTierValidityPeriod>
```

> **Note**
>
> When a quantity range is configured to consume a granted allowance based on a balance element, ECE normally uses the available balance for that balance element. However, if the usage request includes consumption data, ECE uses the current balance instead of the available balance.

Prorating Recurring Fees and Rollovers

When creating charge offers in XML, you use the following XML elements to specify how to apply cycle fees and rollovers when customers purchase, cancel, or change their billing day of month (DOM) in the middle of a billing cycle:

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **<prorateFirst>**: Specifies how to apply cycle fees and rollovers when customers purchase subscriptions in the middle of their billing cycle.
- **<prorateLast>**: Specifies how to apply cycle fees and rollovers when customers cancel subscriptions in the middle of their billing cycle.
- **<prorateCycle>**: Specifies how to apply cycle fees when customers change their billing day of the month (DOM) during any billing cycle.

You can set these XML elements to one of the following:

- **NO_CHARGE**: Do not apply a cycle fee.
- **FULL_CHARGE**: Apply the full cycle fee.
- **PRORATE_CHARGE**: Calculate the cycle fee based on the amount of time the subscription was owned in the billing cycle. This is the default.

The following sample XML entries specify to not apply cycle fees when customers purchase subscriptions in the middle of their billing cycle, to prorate cycle fees when customers cancel subscriptions in the middle of their billing cycle, and to apply full cycle fees when customers change their billing DOM during any billing cycle:

```
<chargeEventMap>
   <eventName>EventDelayedSessionTelcoGsm</eventName>
   <validIfCancelled>false</validIfCancelled>
   <validIfInactive>false</validIfInactive>
   <timezoneMode>EVENT</timezoneMode>
   <minQuantity>1</minQuantity>
   <minQuantityUnit>none</minQuantityUnit>
   <incrementQuantity>1</incrementQuantity>
   <incrementQuantityUnit>none</incrementQuantityUnit>
   <roundingMode>NEAREST</roundingMode>
   <prorateFirst>NO_CHARGE</prorateFirst>
   <prorateLast>PRORATE_CHARGE</prorateLast>
   <prorateCycle>FULL_CHARGE</prorateCycle>
   <chargeRatePlanInfo>
      <targetEngine>ECE</targetEngine>
   </chargeRatePlanInfo>
   <chargeRatePlanName>recurring charge</chargeRatePlanName>
   <ratePlanIID>RatePlanID</ratePlanIID>
</chargeEventMap>
```

For more information, see "About Prorating Recurring Charges and Rollovers".

Setting Full Day Proration

To configure full day proration in XML, you use the **<validityRounding>** and **<scaleRounding>** elements under the **<ChargeOffering>** element.

<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.

Copyright © 2017, 2026, Oracle and/or its affiliates.

<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.
  See "Configuring Timestamp Rounding" in *BRM Configuring and Running Billing*.

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"?>
<pdc:PricingObjectsJXB xmlns:pdc="http://xmlns.oracle.com/communications/
platform/model/pricing">
   <ChargeOffering externalID="67a85a1b-a8af-45a4-8219-e22b085d4a3c">
      <name>CO_OMS-9487</name>
      <internalId>67a85a1b-a8af-45a4-8219-e22b085d4a3c</internalId>
      <pricingProfileName>Product Offering</pricingProfileName>
      <priceListName>Default</priceListName>
      <obsolete>false</obsolete>
      <timeRange>0/inf</timeRange>
      <productSpecName>TelcoGsmTelephony</productSpecName>
      <offerType>SUBSCRIPTION</offerType>
      <priority>0</priority>
      <partial>false</partial>
      <purchaseMin>-1.0</purchaseMin>
      <purchaseMax>-1.0</purchaseMax>
      <ownMin>-1.0</ownMin>
      <ownMax>-1.0</ownMax>
      <applicableQuantity>REMAINING</applicableQuantity>
      <purchaseCycleDOM>0</purchaseCycleDOM>
      <expiryNotification>true</expiryNotification>
      <subscriptionDueNotification>true</subscriptionDueNotification>
      <dateRangeImpactType>EVENT_DATE</dateRangeImpactType>
      <groupSharingEnabled>false</groupSharingEnabled>
      <validityRounding>ON</validityRounding>
      <scaleRounding>ON</scaleRounding>
      <chargeEventMap>
         <eventName>EventBillingProductFeeCycleCycle_forward_monthly</
eventName>
         <validIfCancelled>true</validIfCancelled>
         <validIfInactive>true</validIfInactive>
         <validIfSuspendedActive>true</validIfSuspendedActive>
         <timezoneMode>EVENT</timezoneMode>
         <minQuantity>0</minQuantity>
         <minQuantityUnit>none</minQuantityUnit>
         <incrementQuantity>1</incrementQuantity>
         <incrementQuantityUnit>none</incrementQuantityUnit>
         <roundingMode>NEAREST</roundingMode>
         <prorateFirst>FULL_CHARGE</prorateFirst>
         <prorateLast>PRORATE_CHARGE</prorateLast>
         <prorateCycle>PRORATE_CHARGE</prorateCycle>
         <chargeRatePlanInfo>
            <targetEngine>RRE</targetEngine>
         </chargeRatePlanInfo>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
            <chargeRatePlanName>CRP OMS-9487</chargeRatePlanName>
            <ratePlanIID>5732f8f7-c46d-4fae-aa9a-3d1d5fcf41f7</ratePlanIID>
         </chargeEventMap>
      </ChargeOffering>
</pdc:PricingObjectsJXB>
```

For more information, see "Configuring Full Day Proration".

Setting Taxation for Charges

When configuring charges in XML, you can specify the tax code to use for calculating taxes as well as when to apply taxes. You can also specify a tax selector to determine the appropriate tax code. To do so, set the following elements under **<chargeRatePlan>**:

If you are setting a specific tax code:

- **<taxCode>**: Set this to the name of the tax code.

If you are using a tax selector:

- **<taxSelectorName>**: Set this to the name of a previously defined tax selector.
- **<taxSelectorIID>**: Set this to the internal ID of the tax selector.

In both cases:

- **<taxTime>**: Specify when to apply taxes:
  - **NONE**: Do not apply taxes.
  - **EVENT_TIME**: Taxes are calculated when the event is rated, and they are added to the customer's account balance. This way, you always have an accurate reading of a customer's account balance at any time in the accounting cycle.
  - **BILLING_TIME**: Taxes are calculated when the bill is generated. Deferring tax calculation to the billing process reduces rounding errors because all events of the same type are calculated together. For example, taxes are calculated on the total amount of usage fees rather than on individual usage events.
  - **DYNAMIC**: Taxes are calculated at billing time using the tax rate at the time the event occurred. This provides the benefits of billing-time taxation while allowing you to change tax rates in the middle of a billing cycle. For example, assume a recurring charge's tax rate changes from 2% to 3% on May 15. If a billing cycle ends on May 30, BRM would use the 2% rate to calculate the charge's April 30 through May 14 taxes and the 3% rate to calculate the charge's May 15 through May 29 taxes.

For example, to apply the **VAT** tax code and calculate taxes at billing time, the XML input file would include the following:

```
<chargeRatePlan xmlns:pdc="http://xmlns.oracle.com/communications/platform/
model/pricing">
    <name>Sample Charge</name>
    <pricingProfileName>Subscription</pricingProfileName>
    <priceListName>Default</priceListName>
    <obsolete>false</obsolete>
    <taxCode>VAT</taxCode>
    <applicableRums>Occurrence</applicableRums>
    <applicableQuantity>ORIGINAL</applicableQuantity>
    <taxTime>BILLING_TIME</taxTime>
    <todMode>START_TIME</todMode>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
    <applicableQtyTreatment>CONTINUOUS</applicableQtyTreatment>
    <permittedName>TelcoGsmTelephony</permittedName>
    <permittedType>PRODUCT</permittedType>
    <eventName>EventBillingProductFeeCycleCycle_forward_monthly</eventName>
    <cycleFeeFlag>1</cycleFeeFlag>
    <billOffset>0</billOffset>
    ...
</chargeRatePlan>
```

For example, to use the **Sample_TAX_SELECTOR** tax code selector and calculate taxes when the event is rated, the XML input file would include the following:

```
<chargeRatePlan xmlns:pdc="http://xmlns.oracle.com/communications/platform/
model/pricing">
   <name>20220105 $10 Monthly Charge</name>
   <description>$10 Monthly Charge</description>
   <pricingProfileName>Subscription</pricingProfileName>
   <obsolete>false</obsolete>
   <taxSelectorName>Sample_TAX_SELECTOR</taxSelectorName>
   <taxSelectorIID>52407cd9-30f6-44b7-8519-eca5ba196d55</taxSelectorIID>
   <applicableRums>Occurrence</applicableRums>
   <applicableQuantity>ORIGINAL</applicableQuantity>
   <taxTime>EVENT_TIME</taxTime>
   <todMode>START_TIME</todMode>
   <applicableQtyTreatment>CONTINUOUS</applicableQtyTreatment>
   <eventName>EventBillingProductFeeCycleCycle_forward_monthly</eventName>
   <cycleFeeFlag>0</cycleFeeFlag>
   <billOffset>0</billOffset>
. . .
</chargeRatePlan>
```

Using a Tax Exemption Selector

When creating charge offers in XML, you can use the **<taxExemptionSelector>** field to define a tax exemption selector for the charge offer.

To define a tax exemption selector, provide the name of a previously defined tax exemption selector in the **<taxExemptionSelector>** field.

For example, to use the **Sample TaxExemptionSelector** tax exemption selector for a charge offer, the XML input file would include the following:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<pdc:PricingObjectsJXB xmlns:pdc="http://xmlns.oracle.com/communications/
platform/model/pricing">
<chargeOffering xmlns:pdc="http://xmlns.oracle.com/communications/platform/
model/pricing">
   <name>20230105 ChargeOffer - $10 Monthly Charge</name>
   <description>The TelcoGprs offer charges $10 Monthly fee</description>
   <pricingProfileName>Product Offering</pricingProfileName>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
. . .
   <ownMin>-1.0</ownMin>
   <ownMax>-1.0</ownMax>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
   <taxExemptionSelectorName>Sample TaxExemptionSelector</
taxExemptionSelectorName>
   <applicableQuantity>REMAINING</applicableQuantity>
   <purchaseCycleDOM>0</purchaseCycleDOM>
. . .
</chargeOffering>
</pdc:PricingObjectsJXB>
```

Sharing Products Automatically with Sharing Group Members

When creating charge offers in XML, you can specify whether the charge offer is automatically shared with all members of an automated product sharing group when it is purchased by the group's owner. To do so, you use the **<groupSharingEnabled>** element under the **<chargeOffering>** element.

Set the **<groupSharingEnabled>** element to one of the following:

- **true**: The charge offer is automatically shared with all members of an automated product sharing group when it is purchased by the group's owner.
- **false**: The charge offer is not automatically shared with members of an automated product sharing group.

The following XML example specifies to automatically share the **co1** charge offer with members of an automated product sharing group:

```
<chargeOffering xmlns:pdc="http://xmlns.oracle.com/communications/platform/
model/pricing" externalID="ExternalID">
   <name>co1</name>
   <internalId>InternalID</internalId>
   <pricingProfileName>Product Offering</pricingProfileName>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <timeRange>0/inf</timeRange>
   <productSpecName>Ip</productSpecName>
   <offerType>SUBSCRIPTION</offerType>
   <priority>0</priority>
   <partial>false</partial>
   <purchaseMin>-1.0</purchaseMin>
   <purchaseMax>-1.0</purchaseMax>
   <ownMin>-1.0</ownMin>
   <ownMax>-1.0</ownMax>
   <applicableQuantity>REMAINING</applicableQuantity>
   <purchaseCycleDOM>0</purchaseCycleDOM>
   <expiryNotification>true</expiryNotification>
   <subscriptionDueNotification>true</subscriptionDueNotification>
   <dateRangeImpactType>EVENT_DATE</dateRangeImpactType>
   <groupSharingEnabled>true</groupSharingEnabled>
   <validityRounding>NOT_SET</validityRounding>
   <scaleRounding>OFF</scaleRounding>
   <chargeEventMap>
      <eventName>EventSession</eventName>
      <validIfCancelled>true</validIfCancelled>
      <validIfInactive>true</validIfInactive>
      <validIfSuspendedActive>true</validIfSuspendedActive>
      <timezoneMode>EVENT</timezoneMode>
      <minQuantity>1</minQuantity>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
<minQuantityUnit>none</minQuantityUnit>
<incrementQuantity>1</incrementQuantity>
<incrementQuantityUnit>none</incrementQuantityUnit>
<roundingMode>NEAREST</roundingMode>
<prorateFirst>PRORATE_CHARGE</prorateFirst>
<prorateLast>PRORATE_CHARGE</prorateLast>
<prorateCycle>PRORATE_CHARGE</prorateCycle>
<chargeRatePlanInfo>
   <targetEngine>ECE</targetEngine>
</chargeRatePlanInfo>
...
```

Using Date Ranges for Versioning

When creating charge offers in XML, you can add balance impacts with new date ranges to create new versions of the same charge offer.

You use the **<dateRangeImpactType>** field on charge offers to specify whether existing subscriptions move to the new balance impacts or continue with the old balance impacts.

BRM determines which balance impacts to apply based on the value of **<dateRangeImpactType>** as follows:

- **EVENT_DATE**: The charge event date determines which balance impacts are applied. Any existing subscriptions will use the new balance impacts after the existing date range ends. This is the default behavior.
- **PURCHASE_DATE**: The original purchase date determines which balance impacts are applied. Any existing subscriptions will continue to use the old balance impacts. Any new subscriptions purchased within the new date range will use the new balance impacts.

> **Note**
>
> If a customer transfers their subscription as part of a service group transfer and **<dateRangeImpactType>** is set to **PURCHASE_DATE**, the balance impacts will be determined based on the service transfer date, not the original purchase date.

- **INSTANTIATED_DATE**: The original service instantiation date determines which balance impacts are applied. Any existing subscriptions will continue to use the old balance impacts. Any new subscriptions instantiated during the new date range will use the new balance impacts. 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 pricing, while those who purchase under the old pricing can take advantage of the new pricing if it comes into effect before their new service is available.

The following shows an example of a charge offer using **<dateRangeImpactType>**:

```
<ChargeOffering externalID="1234">
    <name>Charge Offering With Purchase Date Impact</name>
    <internalId>InternalID</internalId>
    <pricingProfileName>Product Offering</pricingProfileName>
    <priceListName>Default</priceListName>
    <obsolete>false</obsolete>
    <timeRange>0/inf</timeRange>
    <productSpecName>TelcoGsmTelephony</productSpecName>
    <dateRangeImpactType>PURCHASE_DATE</dateRangeImpactType>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
    <offerType>SUBSCRIPTION</offerType>
    <priority>0</priority>
    <partial>false</partial>
    <purchaseMin>-1.0</purchaseMin>
    <purchaseMax>-1.0</purchaseMax>
    <ownMin>-1.0</ownMin>
    <ownMax>-1.0</ownMax>
    <applicableQuantity>REMAINING</applicableQuantity>
    <chargeEventMap>
        ...
    </chargeEventMap>
</ChargeOffering>
```

The following examples describe how the value of **<dateRangeImpactType>** affects the balance impacts applied. In the examples, you have an existing charge offer with a charge with a credit balance impact of 1 GB of free data every week. You add a new balance impact crediting 1.5 GB, set the end date of the old balance impact to June 10th, and set the start date for the new balance impact to June 11th. Depending on the value of **<dateRangeImpactType>** for the charge offer, the following happens:

- If **<dateRangeImpactType>** is set to **EVENT_DATE**, all subscriptions get 1 GB until June 10th, and 1.5 GB thereafter.
- If **<dateRangeImpactType>** is set to **PURCHASE_DATE**, subscriptions purchased before June 10th continue to get 1 GB, even for events occurring after June 10th. Subscriptions purchased after June 10th get 1.5 GB.
- If **<dateRangeImpactType>** is set to **INSTANTIATED_DATE**, subscriptions instantiated before June 10th continue to get 1 GB, even for events occurring after June 10th. Subscriptions instantiated after June 10th get 1.5 GB, even if they were purchased before.

Applying Price Tags to Charges

When creating charges in XML, you can apply price tags to them to override the prices at buying time.

To do so, first you must define the price tag setup components. The following example shows a price tag configuration object. See "Price Tag Setup Component" for information about the tags used in this element and "Configuring Price Tags" for information about creating price tag setup components.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<cim:ConfigObjects xmlns:cim="http://xmlns.oracle.com/communications/platform/
model/Config">
      <priceTagConfigs>
        <name>ChargeTag</name>
        <description>A sample price tag.</description>
        <priceListName>Default</priceListName>
       <tags>
            <ruleType>LIST</ruleType>
            <ruleValue>10;20;30;40;50</ruleValue>
            <resourceId>840</resourceId>
            <resourceUnitType>ANY</resourceUnitType>
            <permittedNames>
                    <name>TelcoGsmSms</name>
                    <applicableToAllChildServices>false</
applicableToAllChildServices>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
            </permittedNames>
         </tags>
        </priceTagConfigs>
</cim:ConfigObjects>
```

After submitting the setup components, you can add price tags to charge offers by defining the **<priceTag>** element under the charge with these elements:

- **<attributeName>**: The name of the attribute the tag applies to. This value of is dynamically determined during rating.
- **<tagName>**: The price tag name. This must match the name of a price tag setup component. See "Getting Information about Price Tags" in *BRM Opcode Guide* for information about retrieving the list of price tags.
- **<tagScope>**: The charging profile for the price tag. Can be one of the following:
  - EVENT_PROFILE
  - PRODUCT_PROFILE
  - CUSTOMER_PROFILE

> **Note**
>
> Only the **EVENT_PROFILE** tag scope is supported for charges with the Convergent Usage pricing profile.

- **<description>**: The price tag's description. This is independent from the description defined for the price tag setup component. You can use it to be more precise, and defining this description will not make any changes to the description of the setup component.

Example: Price tag in a balance impact

The following example shows a price tag configured for a usage event at the balance impact level.

```
<fixedCharge>
     <price>35.0</price>
     <unitOfMeasure>NONE</unitOfMeasure>
     <balanceElementNumCode>840</balanceElementNumCode>
     <discountable>true</discountable>
     <priceType>CONSUMPTION</priceType>
     <priceTag>
          <attributeName>price</attributeName>
          <tagName>ChargeTag</tagName>
          <tagScope>EVENT_PROFILE</tagScope>
          <description>Price tag applied to the fixed charge</description>
     </priceTag>
 </fixedCharge>
```

The configuration in the example is validated by comparing values for the following elements:

- **<tagName>** in the charge to the list of price tag names in setup components.
- **<balanceElementNumCode>** in the charge to **<resourceId>** in the ChargeTag setup component

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **<unitOfMeasure>** in the charge to **<resourceUnitType>** in the ChargeTag setup component
- **<permittedName>** in the charge (not shown in this example) to **<permittedNames>** in the ChargeTag setup component

In the example, the default price is **35.0**. At purchase time, you could choose from the values configured for the price tag setup component (10, 20, 30, 40, or 50). If no value is provided, the default price of 35.0 is used.

Example: Price tags in a quantity range

The following example shows different price tags configured for different quantity ranges within a price tier, as well as at the top-level price tier. The configuration in the example is validated in the same way as described in the previous example.

```
<priceTierValidityPeriod>
  <lowerBound>0</lowerBound>
  <validFrom>0</validFrom>
  <priceTierRange>
    <upperBoundExpression>
      <numberTBExpression>
        <value>5</value>
      </numberTBExpression>
    </upperBoundExpression>
    <scaledCharge>
      <price>10.0</price>
      <unitOfMeasure>NONE</unitOfMeasure>
      <balanceElementNumCode>840</balanceElementNumCode>
      <discountable>true</discountable>
      <priceType>CONSUMPTION</priceType>
      <incrementStep>1.0</incrementStep>
      <incrementRounding>NONE</incrementRounding>
    </scaledCharge>
    <priceTag>
      <attributeName>upperBound</attributeName>
      <priceTagId>0</priceTagId>
      <tagName>ptconfig2</tagName>
      <tagScope>EVENT_PROFILE</tagScope>
    </priceTag>
  </priceTierRange>
  <priceTierRange>
    <upperBoundExpression>
      <numberTBExpression>
        <value>NO_MAX</value>
      </numberTBExpression>
    </upperBoundExpression>
    <scaledCharge>
      <price>20.0</price>
      <unitOfMeasure>NONE</unitOfMeasure>
      <balanceElementNumCode>840</balanceElementNumCode>
      <discountable>true</discountable>
      <priceType>CONSUMPTION</priceType>
      <incrementStep>1.0</incrementStep>
      <incrementRounding>NONE</incrementRounding>
    </scaledCharge>
    <priceTag>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
      <attributeName>upperBound</attributeName>
      <priceTagId>0</priceTagId>
      <tagName>universal</tagName>
      <tagScope>EVENT_PROFILE</tagScope>
    </priceTag>
  </priceTierRange>
  <priceTag>
    <attributeName>lowerBound</attributeName>
    <priceTagId>0</priceTagId>
    <tagName>ptconfig1</tagName>
    <tagScope>EVENT_PROFILE</tagScope>
  </priceTag>
</priceTierValidityPeriod>
```

Specifying Validity Period to Start from Event Occurrence Date

When creating charges in XML, you can specify the validity period of noncurrency balances to start from the date of the event's occurrence and end one cycle later by doing the following:

- Setting **<startValidityMode>** to **IMMEDIATE**
- Setting **<endValidityMode>** to **RELATIVE_TO_START**
- Setting **<validityRange>** to **0/inf**
- Setting **<relativeStartOffset>** to **-1**
- Setting **<relativeEndOffset>** to **0**
- Setting **<relativeEndOffsetUnit>** to **EVENT_CYCLE**

The following sample shows how to start the validity period from the event's occurrence and end one cycle later:

```
<tierRange>
         <upperBound>NO_MAX</upperBound>
         <recurringCharge>
               <price>-500.0</price>
               <unitOfMeasure>MINUTE</unitOfMeasure>
               <balanceElementNumCode>1020102</balanceElementNumCode>
               <discountable>true</discountable>
               <priceType>GRANT</priceType>
               <priceValidity>
                    <startValidityMode>IMMEDIATE</startValidityMode>
                    <endValidityMode>RELATIVE_TO_START</endValidityMode>
                    <validityRange>0/inf</validityRange>
                    <relativeStartOffset>-1</relativeStartOffset>
                    <relativeEndOffset>0</relativeEndOffset>
                    <relativeEndOffsetUnit>EVENT_CYCLE</relativeEndOffsetUnit>
                </priceValidity>
                <proratable>true</proratable>
                <impactType>SCALED</impactType>
           </recurringCharge>
</tierRange>
```

Configuring the Automatic Renewal of Allowances

When configuring charge rate plans in XML, you can specify to renew allowances automatically by using the following XML elements under **<AutoRenewResourceExpression>**. To find the value for non-currency resources in PDC, go

Copyright © 2017, 2026, Oracle and/or its affiliates.

to the **Setup** section of the navigation pane. Click **Balance Elements** under **Pricing Elements**. In the table, look for the ID of the resource you want to use.

- **<renewalLimitResource>**: The numeric code of the non-currency balance element that tracks the maximum number of renewals. In PDC, this numeric code is higher than 10,000.
- **<renewalResource>**: Defines the balance element to be renewed.
- **<renewalAmountLimitResource>**: The numeric code of the non-currency balance element that tracks the maximum currency amount allowed for renewals. Use a separate tracker balance element rather than the primary allowance balance element.
- **<maxCapResource>**: The numeric code of the non-currency balance element that defines the maximum limit for granting a resource. For example, if the balance element is megabytes, the maxCapResource will be the numeric code for megabytes. The credit limit is -INF to +INF. Use a separate tracker balance element (with credit limit set to +INF) as the maxCapResource in the renewal Quantity Range under the same usage event, rather than the primary allowance balance element.
- **<maxCapResourceQuantity>**: Defines the quantity of the max cap resource.

> **Note**
>
> In the **charging-settings** XML file, set **saveNonCounterGrantDuringOngoingSession="true"** so the grant is posted during active sessions.

The following sample shows how to configure the automatic renewal of allowances:

```
<chargeRatePlan>
    ...
    ...
    <priceTierValidityPeriod>
        ...
        ...
            <priceTierRange>
                <upperBoundExpression>
                    <autoRenewResourceExpression>
                        <renewalLimitResource>1000100</renewalLimitResource>
                        <renewalResource>100002</renewalResource>
                        <renewalAmountLimitResource>1000101</
renewalAmountLimitResource>
                        <maxCapResource>840</maxCapResource>
                        <maxCapResourceQuantity>100</maxCapResourceQuantity>
                    </autoRenewResourceExpression>
                </upperBoundExpression>
                <scaledCharge>
                    <price>2.0</price>
                    <unitOfMeasure>NONE</unitOfMeasure>
                    <balanceElementNumCode>840</balanceElementNumCode>
                    <discountable>true</discountable>
                    <priceType>CONSUMPTION</priceType>
                    <incrementStep>1.0</incrementStep>
                    <incrementRounding>NONE</incrementRounding>
                </scaledCharge>
            </priceTierRange>
        ...
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
        ...
    </priceTierValidityPeriod>
    ...
    ...
</chargeRatePlan>
```

Discount Offer Configuration Examples

To configure discount offer functionality by using the PDC XML Pricing Interface, see the following examples:

- Consuming Noncurrency Resources for Discounts in Order of Expiration
- Sharing Discount Offers Automatically with Sharing Group Members
- Marking Discount Offers as Eligible for Tying with Charge Offers in XML
- Using a Tax Selector
- Configuring Discount Validity Rounding
- Applying Price Tags to Discounts
- Using Instant Debit in Discount Balance Impacts
- Marking Discount Offers as Eligible for Tying with Charge Offers in XML

Consuming Noncurrency Resources for Discounts in Order of Expiration

When creating discount offers in XML for rating in Oracle Communications Billing and Revenue Management Elastic Charging Engine (ECE), you can specify whether to consume noncurrency resources for the discounts in order of earliest expiry, or in order of any discount rules configured.

To do so, set the **<resourcesConsumptionOrder>** element under the discount (**alterationRatePlan**) to one of the following:

- **EARLIEST_EXPIRY**: Consume noncurrency resources according to which resource expires first. For example, if the customer is subscribed to one discount that grants minutes that expire monthly and another that grants minutes that expire yearly, the minutes expiring monthly should be consumed first. If all resources have the same validity dates, the discount rule order will be used.
- **RULE_ORDER**: Consume balances according to the configured discount rules, without taking validity into account.

The following XML example specifies to consume noncurrency resources in order of earliest expiration.

```
<alterationRatePlan xmlns:pdc="http://xmlns.oracle.com/communications/
platform/model/pricing">
    <name>Consumption Order Example Discount</name>
    <internalId>InternalID</internalId>
    <pricingProfileName>Standard Discount ECE</pricingProfileName>
    <priceListName>Default</priceListName>
    <obsolete>false</obsolete>
    <taxCode>ct_direct</taxCode>
    <resourcesConsumptionOrder>EARLIEST_EXPIRY</resourcesConsumptionOrder>
    <arpDateRange>
        <startDate>0</startDate>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
        <endDate>inf</endDate>
        <alterationConfiguration>
            <applicableChargeAndQuantity>REMAINING_CHARGE</
applicableChargeAndQuantity>
            <arpCompositePopModel>
                <name>Pricing</name>
                <alterationPopModel>
                    <priceTier>
                        <distributionMethod>FROM_BAL_IMPACT</
distributionMethod>
                        <tierBasis>
                            <chargeTierExpression>
                                <useTierComponent>false</useTierComponent>
                                <preRated>false</preRated>
                            </chargeTierExpression>
                        </tierBasis>
                        <lowerBound>
                            <numberTBExpression>
                                <value>0</value>
                            </numberTBExpression>
                        </lowerBound>
                        <tierRange>
                            <upperBound>
                                <numberTBExpression>
                                    <value>NO_MAX</value>
                                </numberTBExpression>
                            </upperBound>
                            <percentAlteration>
                                <price>-0.11</price>
                                <unitOfMeasure>NONE</unitOfMeasure>
                                <balanceElementNumCode>840</
balanceElementNumCode>
                                <alterationAppliesTo>USER</
alterationAppliesTo>
                                <alterationBasedOn>
                                    <chargeBasisExpression>
                                        <useTierComponent>false</
useTierComponent>
                                        <preRated>false</preRated>
                                    </chargeBasisExpression>
                                </alterationBasedOn>
                                <priceType>GRANT</priceType>
                                <priceValidity>
                                    <startValidityMode>IMMEDIATE</
startValidityMode>
                                    <endValidityMode>NEVER</endValidityMode>
                                    <validityRange>0/inf</validityRange>
                                    <relativeStartOffset>-1</
relativeStartOffset>
                                    <relativeEndOffset>-1</relativeEndOffset>
                                </priceValidity>
                            </percentAlteration>
                        </tierRange>
                    </priceTier>
                </alterationPopModel>
            </arpCompositePopModel>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
        </alterationConfiguration>
    </arpDateRange>
</alterationRatePlan>
```

Sharing Discount Offers Automatically with Sharing Group Members

When creating discount offers in XML, you can specify whether the discount offer is automatically shared with all members of an automated discount offer sharing group when it is purchased by the group's owner. To do so, you use the **<groupSharingEnabled>** element under the **<alterationOffering>** element.

Set the **<groupSharingEnabled>** element to one of the following:

- **true**: The discount offer is shared with all members of an automated discount offer sharing group when it is purchased by the group's owner.
- **false**: The discount offer is not shared with members of an automated discount offer sharing group.

The following XML example specifies to automatically share the **do1** discount offer with members of an automated discount offer sharing group:

```
<alterationOffering xmlns:pdc="http://xmlns.oracle.com/communications/
platform/model/pricing" externalID="ExternalID">
   <name>do1</name>
   <internalId>InternalID</internalId>
   <pricingProfileName>Product Offering</pricingProfileName>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <timeRange>0/inf</timeRange>
   <productSpecName>Ip</productSpecName>
   <offerType>SUBSCRIPTION</offerType>
   <priority>0</priority>
   <purchaseMin>-1.0</purchaseMin>
   <purchaseMax>-1.0</purchaseMax>
   <ownMin>-1.0</ownMin>
   <ownMax>-1.0</ownMax>
   <applicableChargeAndQuantity>REMAINING_CHARGE</
applicableChargeAndQuantity>
   <groupSharingEnabled>true</groupSharingEnabled>
   ...
```

Using a Tax Selector

You can specify a tax selector for both a subscription and for a convergent pricing profile (for ECE).

You can specify the tax selector for a subscription at both the discount (**alterationRatePlan**) level and the tierRange level. In both cases, you use the **<taxSelectorName>** parameter to set the tax selector.

Copyright © 2017, 2026, Oracle and/or its affiliates.

For example, to use the **Sample_TAX_SELECTOR** tax selector for a subscription discount and the **Sample_TAX_SELECTOR_2** tax selector for a tierRange, the XML input file would include the following:

```
<alterationRatePlan xmlns:pdc="http://xmlns.oracle.com/communications/
platform/model/pricing">
   <name>20230302-0855 DiscountRatePlan</name>
   <pricingProfileName>Subscription Discount</pricingProfileName>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <taxSelectorName>Sample_TAX_SELECTOR</taxSelectorName>
   <arpDateRange>
      <startDate>0</startDate>
      <endDate>inf</endDate>
      <alterationConfiguration>
         <applicableChargeAndQuantity>REMAINING_CHARGE</
applicableChargeAndQuantity>
         <arpCompositePopModel>
            <name>Pricing</name>
            <alterationPopModel>
               <priceTier>
. . .
                  <tierRange>
. . .
                     <percentAlteration>
. . .
                        <priceType>CONSUMPTION</priceType>
                        <taxTime>EVENT_TIME</taxTime>
                        <taxSelectorName>Sample_TAX_SELECTOR_2</
taxSelectorName>
                     </percentAlteration>
. . .
                  </tierRange>
               </priceTier>
            </alterationPopModel>
         </arpCompositePopModel>
      </alterationConfiguration>
   </arpDateRange>
</alterationRatePlan>
```

You specify the tax selector for a convergent pricing profile (for ECE) at the discount (**alterationRatePlan**) level using the **<taxSelectorName>** parameter.

For example, to use the **Sample_TAX_SELECTOR** tax selector for a convergent pricing profile discount, the XML input file would include the following:

```
<alterationRatePlan xmlns:pdc="http://xmlns.oracle.com/communications/
platform/model/pricing">
   <name>20230227 - DiscountRatePlan</name>
   <pricingProfileName>Standard Discount</pricingProfileName>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <taxSelectorName>Sample_TAX_SELECTOR</taxSelectorName>
   <resourcesConsumptionOrder>RULE_ORDER</resourcesConsumptionOrder>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
. . .
</alterationRatePlan>
```

Configuring Discount Validity Rounding

When creating discount offers in XML, you can specify whether a discount offer's validity period starts when the discount is purchased or at midnight (00:00:00) of the day it was purchased.

To do so, set the <**validityRounding**> element under the <**alterationOffering**> 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 (00:00:00) of the day the discount is purchased.
- **NOT_SET**: Uses the systemwide setting. This is the default.

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" standalone="yes"?><pdc:PricingObjectsJXB
xmlns:pdc="http://xmlns.oracle.com/communications/platform/model/
pricing"><alterationOffering xmlns:pdc="http://xmlns.oracle.com/
communications/platform/model/pricing" externalID="10904d0d-9636-4450-9dfe-
f9152c7ac39f">
   <name>DO_OMS_16916</name>
   <internalId>10904d0d-9636-4450-9dfe-f9152c7ac39f</internalId>
   <pricingProfileName>Product Offering</pricingProfileName>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <timeRange>0/inf</timeRange>
   <productSpecName>TelcoGsmTelephony</productSpecName>
   <offerType>SUBSCRIPTION</offerType>
   <priority>0</priority>
   <purchaseMin>-1.0</purchaseMin>
   <purchaseMax>-1.0</purchaseMax>
   <ownMin>-1.0</ownMin>
   <ownMax>-1.0</ownMax>
   <applicableChargeAndQuantity>REMAINING_CHARGE</applicableChargeAndQuantity>
   <groupSharingEnabled>false</groupSharingEnabled>
   <validityRounding>ON</validityRounding>
   <alterationEventMap>
      <eventName>EventBillingProductFeeCycleCycle_forward_monthly</eventName>
      <validIfCancelled>true</validIfCancelled>
      <validIfInactive>true</validIfInactive>
      <snowball>false</snowball>
      <validAtStartNotValidAtEnd>PRORATE_DISCOUNT</validAtStartNotValidAtEnd>
      <notValidAtStartValidAtEnd>PRORATE_DISCOUNT</notValidAtStartValidAtEnd>
      <notValidAtStartNotValidAtEnd>PRORATE_DISCOUNT</
notValidAtStartNotValidAtEnd>
      <alterationRatePlanInfo>
         <targetEngine>BRE</targetEngine>
      </alterationRatePlanInfo>
      <alterationRatePlanName>d_jan3_24_1516</alterationRatePlanName>
      <ratePlanIID>e861fb12-b4ae-4b53-9ef5-00dfce532047</ratePlanIID>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
   </alterationEventMap>
</alterationOffering></pdc:PricingObjectsJXB>
```

Applying Price Tags to Discounts

When creating discounts in XML, you can apply price tags to them to override the prices at buying time.

To do so, first you must define the price tag setup components. The following example shows a price tag configuration object. See "Price Tag Setup Component" for information about the tags used in this element and "Configuring Price Tags" for information about creating price tag setup components.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<cim:ConfigObjects xmlns:cim="http://xmlns.oracle.com/communications/platform/
model/Config">
      <priceTagConfigs>
        <name>DiscountTag</name>
        <description>A sample price tag for discounts.</description>
        <priceListName>Default</priceListName>
       <tags>
            <ruleType>RANGE</ruleType>
            <ruleValue>-.05;-.20</ruleValue>
            <resourceId>840</resourceId>
            <resourceUnitType>ANY</resourceUnitType>
            <permittedNames>
                    <name>TelcoGsmSms</name>
                    <applicableToAllChildServices>false</
applicableToAllChildServices>
            </permittedNames>
        </tags>
       </priceTagConfigs>
</cim:ConfigObjects>
```

After submitting the setup components, you can add price tags to discount offers by defining the **<priceTag>** element under the discount with these elements:

- **<attributeName>**: The name of the attribute the tag applies to. This value of is dynamically determined during rating.
- **<tagName>**: The price tag name. This must match the name of a price tag setup component. See "Getting Information about Price Tags" in *BRM Opcode Guide* for information about retrieving the list of price tags.
- **<tagScope>**: The charging profile for the price tag. Can be one of the following:
  - EVENT_PROFILE
  - PRODUCT_PROFILE
  - CUSTOMER_PROFILE

> **Note**
>
> Only the **EVENT_PROFILE** tag scope is supported for charges with the Convergent Usage pricing profile.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **<description>**: The price tag's description. This is independent from the description defined for the price tag setup component. You can use it to be more precise, and defining this description will not make any changes to the description of the setup component.

The following example shows a price tag configured for a discount offer.

```
<percentAlteration>
    <price>-0.10</price>
    <unitOfMeasure>NONE</unitOfMeasure>
    <balanceElementNumCode>840</balanceElementNumCode>
    <alterationAppliesTo>USER</alterationAppliesTo>
    <alterationBasedOn>
        <chargeBasisExpression>
            <useTierComponent>false</useTierComponent>
            <preRated>false</preRated>
        </chargeBasisExpression>
    </alterationBasedOn>
    <priceType>GRANT</priceType>
    <priceValidity>
        <startValidityMode>IMMEDIATE</startValidityMode>
        <endValidityMode>NEVER</endValidityMode>
        <validityRange>0/inf</validityRange>
        <relativeStartOffset>-1</relativeStartOffset>
        <relativeEndOffset>-1</relativeEndOffset>
    </priceValidity>
    <priceTag>
          <attributeName>price</attributeName>
          <tagName>DiscountTag</tagName>
          <description>Price tag applied to the discount.</description>
          <tagScope>EVENT_PROFILE</tagScope>
    </priceTag>
</percentAlteration>
```

The configuration in the example is validated by comparing:

- The **<tagName>** in the discount to the list of price tag names in setup components.
- The **<balanceElementNumCode>** in the discount to the **<resourceId>** in the DiscountTag setup component
- The **<permittedName>** in the discount (not shown in this example) to the **<permittedNames>** in the DiscountTag setup component.

In the example, the default discount is **-0.10**. At purchase time, you could enter a value from the range configured for DiscountTag setup component (between -.05 and -.20). If no value is provided for the price tag at purchase time, the default of -.10 is used.

Using Instant Debit in Discount Balance Impacts

When creating discounts in XML, you can specify to consume the balance impact immediately. To do so, you set the following elements under **<fixedAlteration>** or **<percentAlteration>**:

- **price:** The amount to instant debit. This must be a positive value.
- **priceType:** Type of impact: **grant**.
- **startValidityMode:** Defines start validity mode:
  - **FOREVER**

Copyright © 2017, 2026, Oracle and/or its affiliates.

The following shows an example for creating an **Instant Debit** impact type which instantly debits a price of 333 US Dollars and 3 Cents from the customer's balance (instead of being reserved) using **AlterationRatePlans**:

```
<fixedAlteration>
     <price>333.3</price>
     <unitOfMeasure>NONE</unitOfMeasure
     <balanceElementNumCode>578</balanceElementNumCode>
     <alterationAppliesTo>USER</alterationAppliesTo>
     <alterationBasedOn>
          <chargeBasisExpression>
               <useTierComponent>false</useTierComponent>
               <preRated>false</preRated>
          </chargeBasisExpression>
     </alterationBasedOn>
     <priceType>GRANT</priceType>
     <priceValidity>
          <startValidityMode>FOREVER</startValidityMode>
          <endValidityMode>NEVER</endValidityMode>
          <validityRange>0/inf</validityRange>
          <relativeStartOffset>-1</relativeStartOffset>
          <relativeEndOffset>-1</relativeEndOffset>
     </priceValidity>
     <incrementStep>0.0</incrementStep>
     <prorateLastIncrementStep>true</prorateLastIncrementStep>
</fixedAlteration>
```

Marking Discount Offers as Eligible for Tying with Charge Offers in XML

When creating or updating discount offers using the PDC XML Pricing Interface, you can make a discount offer eligible for tying with charge offers at purchase time. To do so, set the **<tieToChargeOfferEnabled>** element under the **<alterationOffering>** element to **true**.

Set **<tieToChargeOfferEnabled>** to **false** or omit the element to leave the discount offer not eligible for tying with specific charge offers.

Example

```
<alterationOffering xmlns:pdc="http://xmlns.oracle.com/communications/
platform/model/pricing" externalID="ExternalID">
  <name>SIM Activation Discount</name>
  <internalId>InternalID</internalId>
  <pricingProfileName>Product Offering</pricingProfileName>
  <priceListName>Default</priceListName>
  <obsolete>false</obsolete>
  <timeRange>0/inf</timeRange>
  <productSpecName>TelcoGsmTelephony</productSpecName>
  <offerType>SUBSCRIPTION</offerType>
  <priority>0</priority>
  <purchaseMin>-1.0</purchaseMin>
  <purchaseMax>-1.0</purchaseMax>
  <ownMin>-1.0</ownMin>
  <ownMax>-1.0</ownMax>
  <applicableChargeAndQuantity>REMAINING_CHARGE</applicableChargeAndQuantity>
  <tieToChargeOfferEnabled>true</tieToChargeOfferEnabled>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
  <groupSharingEnabled>false</groupSharingEnabled>
  <validityRounding>ON</validityRounding>
  <alterationEventMap>
  . . .
  </alterationEventMap>
</alterationOffering>
```

Package Configuration Examples

To configure package functionality by using the PDC XML Pricing Interface, see the following examples:

- Activating Offers in Packages
- Setting Loan Thresholds for Packages
- Transitioning Packages

Activating Offers in Packages

When creating packages in XML, you can specify whether to activate all charge offers across all bundles in a package when one of the charge offers is activated on first usage. To do so, you use the **<firstUsageActivation>** element under the **<package>** element.

Set the **<firstUsageActivation>** element to one of the following:

- **true**: Activates all charge offers in the package upon first usage of any charge offer in the package.

  For example, assume a package contains bundle 1 with charge offers A and B, and bundle 2 with charge offer C. If the first service a customer uses is in charge offer B, all charge offers (A, B, and C) are activated at that time.

- **false**: When one charge offer in the package is activated on first usage, only that charge offer is activated. The other charge offers are not affected.

  For example, assume a package contains bundle 1 with charge offers A and B, and bundle 2 with charge offer C. If the first service a customer uses is in charge offer B, only charge offer B is activated at that time. This is the default.

The following shows an example of a package with the **<firstUsageActivation>** element set:

```
<package xmlns:pdc="http://xmlns.oracle.com/communications/platform/model/
pricing">
   <name>PackageWithPSCAttributes</name>
   <internalId>InternalID</internalId>
   <pricingProfileName>Subscription</pricingProfileName>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <billOnPurchase>false</billOnPurchase>
   <firstUsageActivation>true</firstUsageActivation>
   <productSpecPackageItem>
      <name>TelcoGsmTelephony</name>
      <balanceSpecificationName>Account Balance Group</
balanceSpecificationName>
      <productSpecName>TelcoGsmTelephony</productSpecName>
      <bundledProductOfferingAssociation>
         <optional>false</optional>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
         <cancelWithService>false</cancelWithService>
         <bundledProductOfferingName>bundleForPkg</bundledProductOfferingName>
      </bundledProductOfferingAssociation>
   </productSpecPackageItem>
   <balanceSpecification>
      <name>Account Balance Group</name>
   </balanceSpecification>
</package>
```

Setting Loan Thresholds for Packages

When creating balance credit profiles for packages in XML, you can use the following types in **<thresholdLevel>** elements to set balance thresholds below which a customer is offered a loan:

- **PERCENTAGE_LOAN**: When this percent of the balance is consumed, the customer is offered a loan. The percentage must be in 5% increments.
- **ABSOLUTE_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, each contained within a separate **<thresholdLevel>** element.

When the customer subscribes to the package, 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 types. The PERCENTAGE_LOAN and ABSOLUTE_LOAN types are used for offering loans, and the PERCENTAGE and ABSOLUTE types are used for different notifications or automatic top-ups:

```
<package>
    <name>Package for Loan Thresholds</name>
    <internalId>InternalID</internalId>
    <pricingProfileName>Subscription</pricingProfileName>
    <priceListName>Default Price List</priceListName>
    <obsolete>false</obsolete>
    <billOnPurchase>false</billOnPurchase>
    <productSpecPackageItem>
        <name>TelcoGsmTelephony</name>
        <balanceSpecificationName>Account Balance Group</
balanceSpecificationName>
        <productSpecName>TelcoGsmTelephony</productSpecName>
        <bundledProductOfferingAssociation>
            <optional>false</optional>
            <cancelWithService>false</cancelWithService>
            <bundledProductOfferingName>Bundle for Voice</
bundledProductOfferingName>
        </bundledProductOfferingAssociation>
        </productSpecPackageItem>
    <balanceSpecification>
        <name>Account Balance Group</name>
        <balanceElementSpecification>
            <balanceElementNumCode>840</balanceElementNumCode>
            <consumptionRule>EARLIEST_EXPIRATION</consumptionRule>
            <balanceCreditProfile>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
                    <floor>10.0</floor>
                    <ceiling>100.0</ceiling>
                    <thresholdLevel>
                        <threshold>90.0</threshold>
                        <type>PERCENTAGE</type>
                    </thresholdLevel>
                    <thresholdLevel>
                        <threshold>25.0</threshold>
                        <type>ABSOLUTE</type>
                    </thresholdLevel>
                    <thresholdLevel>
                        <threshold>95.0</threshold>
                        <type>PERCENTAGE_LOAN</type>
                    </thresholdLevel>
                    <thresholdLevel>
                        <threshold>5.0</threshold>
                        <type>ABSOLUTE_LOAN</type>
                    </thresholdLevel>
                </balanceCreditProfile>
            </balanceElementSpecification>
        </balanceSpecification>
</package>
```

Transitioning Packages

When creating package transitions in XML, you can configure how PDC applies charges when customers transition packages in the middle of their billing cycle by using the **<prorationType>** element under the **<packageTransition>** element.

Set the **<prorationType>** element to one of the following:

- **PRORATE_CHARGE**: Specifies to prorate charges for both packages. This is the default.
- **ORIGINAL_CHARGE**: Specifies to apply the full charges from the original package.
- **TRANSFER_CHARGE**: Specifies to apply the full charges from the new package.

The following shows an example that applies the full charge from the original package when the package is transitioned in the middle of a billing cycle:

```
<packageTransition xmlns:pdc="http://xmlns.oracle.com/communications/platform/
model/pricing">
   <name>sample package transition</name>
   <internalId>InternalID</internalId>
   <pricingProfileName>Subscription</pricingProfileName>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <active>true</active>
   <timeRange>0/inf</timeRange>
   <waivePurchaseFee>true</waivePurchaseFee>
   <waiveCancelFee>true</waiveCancelFee>
   <productSpecName>TelcoGsmTelephony</productSpecName>
   <prorationType>ORIGINAL_CHARGE</prorationType>
   <type>UPGRADE</type>
   <fromPackageName>Package A</fromPackageName>
   <toPackageName>Package B</toPackageName>
</packageTransition>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

Selector Configuration Examples

To configure selector functionality by using the PDC XML Pricing Interface, see the following examples:

- Configuring Tax Selectors
- Configuring Tax Exemption Selectors

Configuring Tax Selectors

You can configure tax selectors, which are used to apply different tax codes based on the value of one or more service, event, profile, or customer attributes. You do this using the **<taxSelector>** element.

The following shows an example of a **<taxSelector>** element:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><pdc:PricingObjectsJXB
xmlns:pdc="http://xmlns.oracle.com/communications/platform/model/pricing">
<taxSelector xmlns:pdc="http://xmlns.oracle.com/communications/platform/model/
pricing">
   <name>Sample_TAX_SELECTOR</name>
   <description>Sample TAX_SELECTOR</description>
   <pricingProfileName>Convergent Usage</pricingProfileName>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <stereoType>TAX_SELECTOR</stereoType>
   <productSpecName>TelcoGsmSms</productSpecName>
   <eventSpecName>EventDelayedSessionTelcoGsm</eventSpecName>
   <validityPeriod>
      <validFrom>0</validFrom>
      <rule>
         <name>Rule1</name>
         <result>
            <resultName>VAT</resultName>
         </result>
         <fieldToValueExpression>
            <operation>REGEX</operation>
            <seperator>;</seperator>
            <fieldName>EventDelayedSessionTelcoGsm.CALLED_ID</fieldName>
            <fieldValue>1800.*</fieldValue>
            <fieldKind>EVENT_SPEC_FIELD</fieldKind>
         </fieldToValueExpression>
      </rule>
      <rule>
         <name>Rule2</name>
         <result>
            <resultName>toll</resultName>
         </result>
         <fieldToValueExpression>
            <operation>REGEX</operation>
            <seperator>;</seperator>
            <fieldName>EventDelayedSessionTelcoGsm.CALLED_ID</fieldName>
            <fieldValue>.*</fieldValue>
            <fieldKind>EVENT_SPEC_FIELD</fieldKind>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
            </fieldToValueExpression>
         </rule>
      </validityPeriod>
      </taxSelector></pdc:PricingObjectsJXB>
```

Configuring Tax Exemption Selectors

You can configure tax exemption selectors, which are used to apply different tax exemptions based on the value of one or more service, event, profile, or customer attributes. You do this using the **<taxExemptionSelector>** element.

The following shows an example of a **<taxExemptionSelector>** element:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<pdc:PricingObjectsJXB xmlns:pdc="http://xmlns.oracle.com/communications/
platform/model/pricing">
<taxExemptionSelector xmlns:pdc="http://xmlns.oracle.com/communications/
platform/model/pricing">
   <name>Sample TaxExemptionSelector</name>
   <description>Sample TaxExemptionSelector</description>
   <pricingProfileName>Product Offering</pricingProfileName>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <stereoType>TAX_EXEMPTION_SELECTOR</stereoType>
   <productSpecName>TelcoGprs</productSpecName>
   <eventSpecName>EventSessionTelcoGprs</eventSpecName>
   <customerSpecName>Account</customerSpecName>
   <validityPeriod>
      <validFrom>0</validFrom>
      <rule>
         <name>TES_rule1</name>
         <result>
            <resultName>USATaxExemptionCode</resultName>
         </result>
         <fieldToValueExpression>
            <operation>EQUAL_TO</operation>
            <seperator>;</seperator>
            <fieldName>TelcoGprs.GPRS_INFO.ROUTING_AREA</fieldName>
            <fieldValue>National</fieldValue>
            <fieldKind>PRODUCT_SPEC_FIELD</fieldKind>
         </fieldToValueExpression>
         <fieldToValueExpression>
            <operation>EQUAL_TO</operation>
            <seperator>;</seperator>
            <fieldName>EventSessionTelcoGprs.TELCO_INFO.PRIMARY_MSID</
fieldName>
            <fieldValue>PMSID10248112193</fieldValue>
            <fieldKind>EVENT_SPEC_FIELD</fieldKind>
         </fieldToValueExpression>
         <fieldToValueExpression>
            <operation>IN_LIST</operation>
            <seperator>;</seperator>

<fieldName>EventBillingProductFeeCycleCycle_forward_monthly.CYCLE_INFO.CYCLE_S
TART_T</fieldName>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
            <fieldValue>20100101</fieldValue>
            <fieldKind>EVENT_SPEC_FIELD</fieldKind>
         </fieldToValueExpression>
         <fieldToValueExpression>
            <operation>IN_LIST</operation>
            <seperator>;</seperator>

<fieldName>EventBillingProductFeeCycleCycle_forward_monthly.CYCLE_INFO.CYCLE_E
ND_T</fieldName>
            <fieldValue>20301231</fieldValue>
            <fieldKind>EVENT_SPEC_FIELD</fieldKind>
         </fieldToValueExpression>
         <fieldToValueExpression>
            <operation>EQUAL_TO</operation>
            <seperator>;</seperator>
            <fieldName>AnyEvent.USAGE_TYPE</fieldName>
            <fieldValue>Downlink</fieldValue>
            <fieldKind>EVENT_SPEC_FIELD</fieldKind>
         </fieldToValueExpression>
         <fieldToValueExpression>
            <operation>EQUAL_TO</operation>
            <seperator>;</seperator>
            <fieldName>Account.ACCOUNT_NO</fieldName>
            <fieldValue>ID12221521</fieldValue>
            <fieldKind>CUSTOMER_SPEC_FIELD</fieldKind>
         </fieldToValueExpression>
      </rule>
      <rule>
         <name>TES_rule2</name>
         <result>
            <resultName>IndiaTaxExemptionCode</resultName>
         </result>
         <fieldToValueExpression>
            <operation>EQUAL_TO</operation>
            <seperator>;</seperator>
            <fieldName>TelcoGprs.GPRS_INFO.ROUTING_AREA</fieldName>
            <fieldValue>SouthWest</fieldValue>
            <fieldKind>PRODUCT_SPEC_FIELD</fieldKind>
         </fieldToValueExpression>
         <fieldToValueExpression>
            <operation>EQUAL_TO</operation>
            <seperator>;</seperator>
            <fieldName>EventSessionTelcoGprs.TELCO_INFO.PRIMARY_MSID</
fieldName>
            <fieldValue>PMSID10265018152</fieldValue>
            <fieldKind>EVENT_SPEC_FIELD</fieldKind>
         </fieldToValueExpression>
         <fieldToValueExpression>
            <operation>IN_LIST</operation>
            <seperator>;</seperator>

<fieldName>EventBillingProductFeeCycleCycle_forward_monthly.CYCLE_INFO.CYCLE_S
TART_T</fieldName>
            <fieldValue>20200101</fieldValue>
            <fieldKind>EVENT_SPEC_FIELD</fieldKind>
         </fieldToValueExpression>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
            <fieldToValueExpression>
               <operation>IN_LIST</operation>
               <seperator>;</seperator>

<fieldName>EventBillingProductFeeCycleCycle_forward_monthly.CYCLE_INFO.CYCLE_E
ND_T</fieldName>
               <fieldValue>20321231</fieldValue>
               <fieldKind>EVENT_SPEC_FIELD</fieldKind>
            </fieldToValueExpression>
            <fieldToValueExpression>
               <operation>EQUAL_TO</operation>
               <seperator>;</seperator>
               <fieldName>AnyEvent.USAGE_TYPE</fieldName>
               <fieldValue>Uplink</fieldValue>
               <fieldKind>EVENT_SPEC_FIELD</fieldKind>
            </fieldToValueExpression>
            <fieldToValueExpression>
               <operation>EQUAL_TO</operation>
               <seperator>;</seperator>
               <fieldName>Account.ACCOUNT_NO</fieldName>
               <fieldValue>ID22220531</fieldValue>
               <fieldKind>CUSTOMER_SPEC_FIELD</fieldKind>
            </fieldToValueExpression>
         </rule>
      </validityPeriod>
</taxExemptionSelector>
</pdc:PricingObjectsJXB>
```

Configuring Impact-Based Candidate Selection by Using the XML Pricing Interface

You can configure a Charge Selector or Generic Selector to select a matching candidate based on its immediate impact to a specified balance element. Set **CandidateSelectionMode** to **MIN_IMPACT** or **MAX_IMPACT**, and set **CandidateSelectionBalanceElementCode** to the balance element that ECE compares. For selection behavior, see "How Impact-Based Candidate Selection Works".

The following example configures a Charge Selector to use **MIN_IMPACT** and compare candidate impacts to the USD balance element:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><pdc:PricingObjectsJXB
xmlns:pdc="http://xmlns.oracle.com/communications/platform/model/
pricing"><chargeRatePlan xmlns:pdc="http://xmlns.oracle.com/communications/
platform/model/pricing">
   <name>c_test_import_cs_1816</name>
   <pricingProfileName>Convergent Usage</pricingProfileName>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <applicableRums>USAGE</applicableRums>
   <applicableQuantity>ORIGINAL</applicableQuantity>
   <taxTime>NONE</taxTime>
   <todMode>START_TIME</todMode>
   <applicableQtyTreatment>CONTINUOUS</applicableQtyTreatment>
   <permittedName>TelcoGsmTelephony</permittedName>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
      <permittedType>PRODUCT</permittedType>
      <eventName>EventDelayedSessionTelcoGsm</eventName>
      <cycleFeeFlag>0</cycleFeeFlag>
      <billOffset>0</billOffset>
      <subscriberCurrency>
         <currencyCode>USD</currencyCode>
         <applicableRum>
            <applicableRumName>USAGE</applicableRumName>
            <minQuantity>0.0</minQuantity>
            <minQuantityUnit>NONE</minQuantityUnit>
            <incrementQuantity>0.0</incrementQuantity>
            <incrementQuantityUnit>NONE</incrementQuantityUnit>
            <roundingMode>NEAREST</roundingMode>
            <crpRelDateRange>
               <absoluteDateRange>
                  <startDate>0</startDate>
                  <endDate>inf</endDate>
               </absoluteDateRange>
               <crpCompositePopModel>
                  <name>Pricing</name>
                  <usageChargePopModel>
                     <priceTier>
                        <distributionMethod>FROM_BAL_IMPACT</distributionMethod>
                        <tierBasis>
                           <rumTierExpression/>
                        </tierBasis>
                        <enforceCreditLimit>DEFAULT</enforceCreditLimit>
                        <rumName>USAGE</rumName>
                        <currencyCode>USD</currencyCode>
                        <priceTierValidityPeriod>
                           <lowerBound>0</lowerBound>
                           <validFrom>0</validFrom>
                           <priceTierRange>
                              <upperBoundExpression>
                                 <numberTBExpression>
                                    <value>NO_MAX</value>
                                 </numberTBExpression>
                              </upperBoundExpression>
                              <scaledCharge>
                                 <price>0.0</price>
                                 <unitOfMeasure>NONE</unitOfMeasure>
                                 <balanceElementNumCode>840</
balanceElementNumCode>
                                 <discountable>true</discountable>
                                 <priceType>CONSUMPTION</priceType>
                                 <incrementStep>1.0</incrementStep>
                                 <incrementRounding>NONE</incrementRounding>
                              </scaledCharge>
                           </priceTierRange>
                        </priceTierValidityPeriod>
                        <applicableQuantity>ORIGINAL</applicableQuantity>
                     </priceTier>
                  </usageChargePopModel>
               </crpCompositePopModel>
            </crpRelDateRange>
         </applicableRum>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
   </subscriberCurrency>
</chargeRatePlan><chargeRateplanSelector xmlns:pdc="http://xmlns.oracle.com/
communications/platform/model/pricing">
   <name>cs_test_import_1816</name>
   <pricingProfileName>Convergent Usage</pricingProfileName>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <stereoType>CHARGE_RATE_PLAN_SELECTOR</stereoType>
   <productSpecName>TelcoGsmTelephony</productSpecName>
   <eventSpecName>EventDelayedSessionTelcoGsm</eventSpecName>
   <customerSpecName>Account</customerSpecName>
   <validityPeriod>
      <validFrom>0</validFrom>
      <rule>
         <name>r1</name>
         <result>
            <resultName>c_test_import_cs_1748</resultName>
            <resultData>
               <key>RateplanIID</key>
               <value>1dc4c380-ba27-43ad-8086-0190a8a5e96e</value>
            </resultData>
         </result>
         <fieldToValueExpression>
            <operation>EQUAL_TO</operation>
            <seperator>;</seperator>
            <fieldName>EventDelayedSessionTelcoGsm.EVENT_NO</fieldName>
            <fieldValue>*</fieldValue>
            <fieldKind>EVENT_SPEC_FIELD</fieldKind>
         </fieldToValueExpression>
      </rule>
   </validityPeriod>
   <modelData>
      <key>ZONING_OPTION</key>
      <value>ZONING_NONE</value>
   </modelData>
   <modelData>
      <key>FIELD_COUNT</key>
      <value>1</value>
   </modelData>
   <modelData>
      <key>RUM</key>
      <value>USAGE</value>
   </modelData>
   <modelData>
      <key>field_0.name</key>
      <value>EventDelayedSessionTelcoGsm.EVENT_NO</value>
   </modelData>
  <modelData>
      <key>field_0.kind</key>
      <value>EVENT_SPEC_FIELD</value>
   </modelData>
   <modelData>
      <key>field_0.valueType</key>
      <value>SINGLE_VALUE</value>
   </modelData>
   <modelData>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
      <key>field_0.defaultValue</key>
      <value>*</value>
   </modelData>
   <modelData>
      <key>field_0.operator</key>
      <value>EQUAL_TO</value>
   </modelData>
   <candidateSelectionMode>MIN_IMPACT</candidateSelectionMode>
   <candidateSelectionBalanceElementCode>INR123</
candidateSelectionBalanceElementCode>
</chargeRateplanSelector></pdc:PricingObjectsJXB>
```

The following example configures a Generic Selector to use **MIN_IMPACT** and compare the immediate impacts of the PremiumQos and StandardQos result branches to the USD balance element:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><pdc:PricingObjectsJXB
xmlns:pdc="http://xmlns.oracle.com/communications/platform/model/
pricing"><genericSelector xmlns:pdc="http://xmlns.oracle.com/communications/
platform/model/pricing">
   <name>gs_test_import_1830</name>
   <pricingProfileName>Convergent Usage</pricingProfileName>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <stereoType>GENERIC_SELECTOR</stereoType>
   <productSpecName>TelcoGsmTelephony</productSpecName>
   <eventSpecName>EventDelayedSessionTelcoGsm</eventSpecName>
   <validityPeriod>
      <validFrom>0</validFrom>
      <rule>
         <name>r1</name>
         <result>
            <resultName>res1</resultName>
         </result>
         <fieldToValueExpression>
            <operation>EQUAL_TO</operation>
            <seperator>;</seperator>
            <fieldName>EventDelayedSessionTelcoGsm.DESCR</fieldName>
            <fieldValue>*</fieldValue>
            <fieldKind>EVENT_SPEC_FIELD</fieldKind>
         </fieldToValueExpression>
      </rule>
   </validityPeriod>
   <modelData>
      <key>FIELD_COUNT</key>
      <value>1</value>
   </modelData>
   <modelData>
      <key>field_0.name</key>
      <value>EventDelayedSessionTelcoGsm.DESCR</value>
   </modelData>
   <modelData>
      <key>field_0.kind</key>
      <value>EVENT_SPEC_FIELD</value>
   </modelData>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
   <modelData>
      <key>field_0.valueType</key>
      <value>SINGLE_VALUE</value>
   </modelData>
   <modelData>
      <key>field_0.defaultValue</key>
      <value>*</value>
   </modelData>
   <modelData>
      <key>field_0.operator</key>
      <value>EQUAL_TO</value>
   </modelData>
   <candidateSelectionMode>FIRST_MATCH</candidateSelectionMode>
   <candidateSelectionBalanceElementCode>NONE</
candidateSelectionBalanceElementCode>
</genericSelector></pdc:PricingObjectsJXB>
```

If matching Generic Selector result branches have the same comparable impact, ECE selects the first matching branch.

Copyright © 2017, 2026, Oracle and/or its affiliates.

33
PDC XML Tags

Learn about various PDC XML tags and what each tag means. This is useful when importing pricing or configuration data to and exporting pricing or configuration data from PDC using the **ImportExportPricing** utility.

For more information, see "About Using the XML Pricing Interface to Create Product Offerings".

Topics in this document:

- Charge Offer
- Discount Offer
- Chargeshare Offer
- Bundle
- Subscription Terms
- Time Model
- Charge Selector
- Generic Selector
- Price Selector
- Discount Selector
- USC Selector
- APN Selector
- Package
- Package List
- Tax Selector
- Tax Exemption Selector
- Price Tag Setup Component

Charge Offer

This section describes the XML tags that you use to configure charge offers in the XML pricing interface.

Charge offers determine the price of one or more events associated with a service. For more information, see "Configuring Charge Offers".

chargeOffering

This is the main tag for configuring a charge offer in PDC. Table 33-1 describes the fields in the **chargeOffering** tag.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-1 chargeOffering

Field Name	Description
name	Unique name of the charge offer. PDC does not allow creation of offerings with duplicate names. It is VARCHAR2 (255).
description	Description of the charge offer. It is VARCHAR2 (255).
internalId	Unique ID (UUID) internally generated by PDC while committing the pricing configuration. It is a unique internal ID for all charge offers, discount offers, bundles, and configurations in PDC and cannot be the same for any other pricing or configuration entity.
applicableQuantity	This can be **REMAINING** or **ORIGINAL** based on the charge offer type.
dateRangeImpactType	When you add balance impacts with new date ranges to an existing charge offer, whether existing subscriptions move to the new balance impact or continue with the old balance impact: • **EVENT_DATE**: The charge event date determines which balance impacts are applied. • **PURCHASE_DATE**: The original purchase date determines which balance impacts are applied. • **INSTANTIATED_DATE**: The original service instantiation date determines which balance impacts are applied. For products that have the first usage attribute for Purchase, this field is set with actual purchase time and will be used to filter the products for events prior to their subscription (for backdated or future dated scenarios). See "Using Date Ranges for Versioning" for information and examples.
expiryNotification	Boolean flag for triggering notifications a few days or weeks before a charge offer is about to expire. See "Specifying Which Expiring Products Trigger Notifications" in *BRM Managing Customers*.
groupSharingEnabled	Boolean flag to enable the charge offer to be shared with all members of a product sharing group when it is purchased by the group's owner. See "Sharing Products Automatically with Sharing Group Members" for information and examples.
obsolete	Boolean flag to obsolete the promoted (or active) objects.
offerType	The type of charge offer: • **Item**: A one-time purchase fee. Only item is created for the charges and there is no purchased product instance for this subscription. • **Subscription**: Charges for any type and combination of events, including one-time, recurring, usage, rollover, remittance, and fold. These charges apply only to the subscriber who owns the charge offer. Item and purchased product instance are created in BRM for this option. • **System**: Charges that apply to all subscribers who use a particular service. These offer types do not require a subscription and are effective to the subscriber at the system level.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-1 (Cont.) chargeOffering

Field Name	Description
ownMax	Specify the maximum number of charge offers that a customer can own at one time. **-1** means unlimited. For example, if a website charge offer provides 100 MB of disk space, you can limit customers to 500 MB by setting the maximum ownership quantity to 5.
ownMin	Specify the minimum number of charge offers that a customer can own at one time. **-1** means unlimited.
partial	Boolean flag to enable partial purchases. This option is used to determine whether customers can purchase only part of a charge offer.
policySpecification	Policy specification is made up of one or more counter policies, each of which defines a gradation in the quality of service (QoS) based on the subscriber's service usage. For example, you can have a policy specification named Platinum for a data service and a balance element for it as Megabytes Used. You can define a counter 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, such as 100-150, 150-200, and 200-No Maximum megabytes.
postExpiryNotification	Boolean flag for triggering notifications a few days or weeks after a charge offer has expired. See "Specifying Which Expiring Products Trigger Notifications" in *BRM Managing Customers*.
postSubscriptionDueNotification	Boolean flag for triggering notifications a few days or weeks after a subscription is past due for renewal. See "Specifying Which Subscription Renewals Trigger Notifications" in *BRM Managing Customers*.
priceListName	This is configured as **Default** by default.
pricingProfileName	The pricing profile determines which features are supported for the charge. PDC determines the pricingProfileName once the charge category and charge type are selected. Possible values: • **Convergent Usage** • **Product Offering** • **Standard ChargeSharing** • **Standard Discount** • **Subscription ChargeSharing** • **Subscription Discount** • **Subscription** To find out the actual pricingProfileName loaded in the system, query the PDC database: `select name, pricingProfileType from aPricingProfile where entityClass = 'PricingProfileDAO' order by pricingProfileType, name;`

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-1 (Cont.) chargeOffering

Field Name	Description
priority	Priority for the charge offer. When more than one charge offer applies to the same event, this priority determines the order in which your billing system considers the charge offers. The higher the number, the higher the priority. **0** is the lowest priority.
productSpecName	The service type to which the charge offer applies.
provisioningTag	Provisioning tag for the charge offer. In BRM, provisioning tags are used to implement extended rating attributes (ERAs), such as a discount for calls to friends and family on a subscriber's birthday, or to configure telco service extensions and supplementary services, such as call forwarding and call blocking.
purchaseCycleDOM	How to align product validity: • *Integer*: Aligns offer validity to a specific day of the month, with valid values from 1 through 31. • **0**: Aligns offer validity to the billing date. See "Aligning Recurring Charges and Product Validity to a Specific Day of the Month" for information and examples.
purchaseMax	Specify the maximum number of charge offers that a customer can buy at one time. **-1** means unlimited.
purchaseMin	Specify the minimum number of charge offers that a customer can buy at one time. **-1** means unlimited.
subscriptionDueNotification	Boolean flag for triggering notifications a few days or weeks before a subscription is due for renewal. See "Specifying Which Subscription Renewals Trigger Notifications" in *BRM Managing Customers*.
TaxExemptionSelectorGroup	This XML tag contains information about the tax exemption selector to be used for the charge offer. See "TaxExemptionSelectorGroup".
taxSupplier	Select a supplier tax ID for the charge offer. This ID specifies the company or corporate division that sells the product and is responsible for collecting taxes for it.
timeRange	The time range during which the offer is available for purchase in the format *startingTime***|***endingTime*. The default is **0/inf**, which specifies a time range from now to never ends.
validityRounding	Specify whether to start the charge offer's validity period at the purchase time or midnight of the purchase day. • **OFF**: Starts at the time of purchase. This overrides the CM **pin.conf** setting at the charge offer level. • **ON**: Starts at midnight (00:00:00) of the day that the charge offer is purchased. This overrides the CM **pin.conf** setting at the charge offer level. • **NOT_SET**: Uses the systemwide setting in the CM **pin.conf** file. This is the default.
scaleRounding	Specify how to calculate the scale: • **OFF**: Calculates it based on the <**validityRounding**> setting. • **ON**: Calculates it based on full days.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-1 (Cont.) chargeOffering

Field Name	Description
ChargeEventMap	The XML tag inside the ChargeOffering (CO) XML tag which illustrates the event details related with the CO. This tag can occur multiple times in the CO tag based on how many events are mapped inside a Single CO. See "chargeEventMap". **Note**: You can map multiple events inside the CO but for override capability, you must only map single type of event under each CO. There will be one-to-one mapping of CO with ChargeEventMap.

TaxExemptionSelectorGroup

The **TaxExemptionSelectorGroup** tag represents the tax exemption selector information for the charge offer. Table 33-2 describes the fields in the **TaxExemptionSelectorGroup** tag.

Table 33-2 TaxExemptionSelectorGroup

Field Name	Description
taxExemptionSelectorName	The name of the tax exemption selector
taxExemptionSelectorIID	The ID of the tax exemption selector

chargeEventMap

The **chargeEventMap** tag is present for each individual event type that a product offering will rate. It maps a given event type to the charging rules for those events. Table 33-3 describes the fields in the **chargeEventMap** tag.

Table 33-3 chargeEventMap

Field Name	Description
eventName	Name of the event. For example, EventDelayedSessionTelcoGsm (**/event/delayed/session/telco/gsm** in BRM) is used for usage events, while EventBillingProductFeeCycleCycle_forward_monthly (**/event/billing/product/fee/cycle/cycle_forward_monthly**) represents a monthly recurring fee (in-advance).
validIfCancelled	Charge even if the instantiated product offering status is canceled. BRM offering statuses are: • 1 = active • 2 = inactive • 3 = canceled
validIfInactive	Charge even if the instantiated product offering status is inactive.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-3 (Cont.) chargeEventMap

Field Name	Description
timezoneMode	Determines the time zone used to rate an event: • From ACCOUNT: Uses the time zone in which the account is registered. • From EVENT: Uses the time zone in which the event occurs. For example, if you make a call from California and your account is registered in New York: • If you select From Account, New York time is used. • If you select From Event, California time is used.
minQuantity	Minimum value that the RUM quantity is rounded to. Value smaller than **minQuantity** is rounded to **minQuantity**. Minimum quantity starts from 0 by default. For example, if an event had a duration of 10 seconds, and the minimum quantity is configured as 30, then 30 seconds would be rated. If the duration of the event were 45 seconds, 45 seconds would be rated.
minQuantityUnit	Unit in which **minQuantity** is expressed. Possible values are: • NONE (**0**) • SECOND (**1**) • MINUTE (**2**) • HOUR (**3**) • DAY (**4**)
incrementQuantity	Incremental value to round RUM Quantity to. RUM Quantity is always rounded to an integral multiple of the Incremental Quantity. For example, if the **incrementQuantity** is configured as 30 for a RUM of seconds, then the amount to rate would be rounded to that increment (see **roundingMode**) and a 43 second call would be rated as 60 seconds (if rounding up) or 30 seconds (if rounding down).
incrementQuantityUnit	Units that **incrementQuantity** is specified in (see "**minQuantityUnit**").
roundingMode	Specifies how RUM quantity is rounded. Possible values are: • Up (**0**) • Down (**1**) • Nearest (**2**)
prorateFirst	How to apply cycle fees and rollovers when customers *purchase* subscriptions in the middle of their billing cycle: • **NO_CHARGE**: Do not apply a cycle fee. • **FULL_CHARGE**: Apply the full cycle fee. • **PRORATE_CHARGE**: Calculate the cycle fee based on the amount of time the subscription was owned in the billing cycle. This is the default.
prorateLast	How to apply cycle fees and rollovers when customers *cancel* subscriptions in the middle of their billing cycle: • **NO_CHARGE**: Do not apply a cycle fee. • **FULL_CHARGE**: Apply the full cycle fee. • **PRORATE_CHARGE**: Calculate the cycle fee based on the amount of time the subscription was owned in the billing cycle. This is the default.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-3 (Cont.) chargeEventMap

Field Name	Description
prorateCycle	How to apply cycle fees when customers change their billing day of the month (DOM) during any billing cycle: • **NO_CHARGE**: Do not apply a cycle fee. • **FULL_CHARGE**: Apply the full cycle fee. • **PRORATE_CHARGE**: Calculate the cycle fee based on the amount of time the subscription was owned in the billing cycle. This is the default.
chargeRatePlanName	Name of the charge configuration inside the CO at a granular and more detailed level. Charge Configuration name can be any name such as OneTime charge, MRC, Rollover, FreeSMS, FreeVoice, and so on.
ratePlanIID	Charge rate plan IID generated internally by PDC system for the chargeEventMap XML tag.
chargeRatePlanInfo	The target engine where the pricing or configuration information is published. ECE means Rating Engine, BRE means batch rating sync to Pipeline component in BRM, and RRE means Realtime sync to BRM system. See "chargeRatePlanInfo".

chargeRatePlanInfo

The **chargeRatePlanInfo** tag represents the target engine where information about the charge offer is to be published. Table 33-4 describes the field in the **chargeRatePlanInfo** tag.

Table 33-4 chargeRatePlanInfo

Field Name	Description
targetEngine	The target engine where the information about this charge offer is to be published. Possible values are: • **BRE** • **ECE** • **RRE** This is automatically taken care of by the **ImportExportPricing** utility.

chargeRatePlan

The root tag for configuring the charge rate plan. This captures details such as the target rating engine, tax code, time model, and so on. Table 33-5 describes the fields in the **chargeRatePlan** tag.

Table 33-5 chargeRatePlan

Field Name	Description
priceListName	This is configured as **Default** by default.
pricingProfileName	**Convergent Usage** for usage events. **Subscription** for one-time and recurring events.
taxCode	The tax code used for the charge. You can use either this parameter or **TaxSelectorGroup**, but not both.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-5 (Cont.) chargeRatePlan

Field Name	Description
TaxSelectorGroup	The name of the tax selector group to use for the charge. You can use either this parameter or **taxCode**, but not both.
applicableRums	Displays the RUMs used in the charge. RUMs specify the units used to measure an event and how to calculate the measurement. In a charge, the pricing for each RUM is contained in a price tier. If a charge uses multiple RUMs, each tier appears in a subtab in the Pricing Details section of the charge tab.
applicableQuantity	Applicable quantity types. Valid values are: • **ORIGINAL** • **REMAINING**
taxTime	When the charge is taxed: • **NONE**: Do not apply taxes. • **EVENT_TIME**: Taxes are calculated when the event is rated, and they are added to the customer's account balance. This way, you always have an accurate reading of a customer's account balance at any time in the accounting cycle. • **BILLING_TIME**: Taxes are calculated when the bill is generated. Deferring tax calculation to the billing process reduces rounding errors because all events of the same type are calculated together. For example, taxes are calculated on the total amount of usage fees rather than on individual usage events. • **DYNAMIC**: Taxes are calculated at billing time using the tax rate at the time the event occurred. This provides the benefits of billing-time taxation while allowing you to change tax rates in the middle of a billing cycle. For example, assume a recurring charge's tax rate changes from 2% to 3% on May 15. If a billing cycle ends on May 30, BRM would use the 2% rate to calculate the charge's April 30 through May 14 taxes and the 3% rate to calculate the charge's May 15 through May 29 taxes.
todMode	Indicates if events should be rated using **startTime**, **endTime**, or split to use both.
applicableQtyTreatment	Indicates if applicable quantity should be treated continuous or isolated.
permittedName	The service to which the charge applies.
permittedType	Specifies whether it is a product or a customer type.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-5 (Cont.) chargeRatePlan

Field Name	Description
cycleFeeFlag	How to align recurring charges: • **0**: Recurring charges are aligned to the billing date. For example, if the billing date is the 1st of the month and the charge offer is purchased on January 10, the charge is prorated and applied on January 10 for the interval January 10 to February 1. For subsequent cycles, the charge is applied on the billing date (for the interval February 1 to March 1, March 1 to April 1, and so on). • **1**: Recurring charges are applied on the product purchase date. Using the previous example, the charge is applied on the 10th of every month (for the interval January 10 to February 10, February 10 to March 10, and so on). See "Aligning Recurring Charges and Product Validity to a Specific Day of the Month" for information and examples.
billOffset	Specifies the offset (in days or months) to bill in advance.
subscriberCurrency	The three letter ISO code for currency.

TaxSelectorGroup

The **TaxSelectorGroup** tag represents the tax selector information for the charge. describes the fields in the **TaxSelectorGroup** tag.

Table 33-6 TaxExemptionSelectorGroup

Field Name	Description
taxSelectorName	The name of the tax selector
taxSelectorIID	The ID of the tax selector

subscriberCurrency

Table 33-7 describes the fields in subscriberCurrency tag.

Table 33-7 subscriberCurrency

Field Name	Description
crpRelDateRange	Charge rate plan relative date range. See "crpRelDateRange".
crpCompositePopModel	Generic pop model. It can be one-time, recurring, or usage type. See "crpCompositePopModel".
applicableRum	RUMs are ways in which you can measure events. The most common RUMs are Duration and Volume. Before you create product offerings, you must define the RUMs available for rating. You define RUMs when you create the service-event map. For each event, you specify the RUMs that can be used for rating it. See "applicableRum".

Copyright © 2017, 2026, Oracle and/or its affiliates.

crpRelDateRange

Table 33-8 describes the relative date range validity of the charge rate plan.

Table 33-8 crpRelDateRange

Field Name	Description
absoluteDateRange	Container for defining the effective date range for a fixed period. See "absoluteDateRange".
relativeDateRange	Container for defining a relative date range for a product purchase. See "relativeDateRange".

absoluteDateRange

Table 33-9 describes the absolute date range validity of the charge rate plan.

Table 33-9 absoluteDateRange

Field Name	Description
startDate	Start date for the fixed period.
endDate	End date for the fixed period.

relativeDateRange

Table 33-10 describes the relative date range details.

Table 33-10 relativeDateRange

Field Name	Description
startOffset	Number of days, hours, minutes, or seconds after the purchase time that the charge takes effect.
startUnit	Days, hours, minutes, or seconds after the purchase time that the charge takes effect.
endOffset	Number of days, hours, minutes, or seconds after the start time that the charge is no longer effective.
endUnit	Days, hours, minutes, or seconds after the start time that the charge is no longer effective.
base	Indicates the base type. Valid value: PRODUCT_PURCHASE

crpCompositePopModel

Table 33-11 describes the fields in the **crpCompositePopModel** tag.

Table 33-11 crpCompositePopModel

Field Name	Description
name	The name of the pricing instance. The default is **Pricing**.
oneTimePopModel	OneTimeProdOfferPriceModel to define the Balance Impacts for OneTime event types. For example, Product Purchase Fee, Product Cancel Fee, and so on. See "oneTimePopModel".

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-11 (Cont.) crpCompositePopModel

Field Name	Description
recurringPopModel	A RecurringProdOfferPriceModel to define the Balance Impacts for Recurring event types. For example, Monthly Cycle Forward fee. See "recurringPopModel".
usageChargePopModel	UsageProdOfferPriceModel to define the Balance Impacts for Usage event types. For example, Telco Gsm Telephony Usage charges. See "usageChargePopModel".

applicableRum

Table 33-12 describes the fields in the **applicableRum** tag.

Table 33-12 applicableRum

Field Name	Description
minQuantity	Minimum value that the RUM quantity is rounded to. Value smaller than **minQuantity** is rounded to **minQuantity**.
minQuantityUnit	The unit for **minQuantity**. Possible values are: • NONE (**0**) • SECOND (**1**) • MINUTE (**2**) • HOUR (**3**) • DAY (**4**)
incrementQuantity	Incremental value to round RUM Quantity to. RUM Quantity is always rounded to an integral multiple of the Incr Quantity.
incrementQuantityUnit	Units that **incrementQuantity** is specified in.
roundingMode	Specifies how RUM quantity is rounded. Possible values are: • Up (**0**) • Down (**1**) • Nearest (**2**)

oneTimePopModel

Table 33-13 describes the field in the **oneTimePopModel** tag.

Table 33-13 oneTimePopModel

Field Name	Description
priceTier	Container object to define the price tier. See "priceTier (OneTimePriceTier)".

priceTier (OneTimePriceTier)

Table 33-14 describes the fields in the **priceTier** tag.

Table 33-14 priceTier (OneTimePriceTier)

Field Name	Description
lowerBound	Starting value for the range.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-14 (Cont.) priceTier (OneTimePriceTier)

Field Name	Description
tierBasis	Container object to define charge tier expression. See "tierBasis".
enforceCreditLimit	Whether to enforce credit limits or let customers exceed them. See "Enforcing or Exceeding Credit Limits" for information and possible values.
tierRange	Container object to define charge tier range. See "tierRange".

tierBasis

Table 33-15 describes the field in the **tierBasis** tag.

Table 33-15 tierBasis

Field Name	Description
AChargeTierExpression	Base abstract entity for Tier Expression.

tierRange

Table 33-16 describes the fields in the **tierRange** tag.

Table 33-16 tierRange

Field Name	Description
oneTimeCharge	Nonrecurring charges, such as setup or cancellation fees. See "oneTimeCharge".
recurringCharge	Ongoing charges that are not generated or affected by usage, such as a monthly subscription fee. See "recurringCharge".

oneTimeCharge

Table 33-17 describes the fields in the **oneTimeCharge** tag.

Table 33-17 oneTimeCharge

Field Name	Description
price	The amount to debit or credit.
unitOfMeasure	The type of unit, such as Minute or Byte, to which the charge applies.
balanceElementNumCode	Numeric code for the balance element.
discountable	Boolean flag to make a balance impact discountable.
priceType	Type of impact: **grant** or **consumption**.
impactType	Used to specify whether the impact is fixed or scaled.

recurringPopModel

Table 33-18 describes the fields in the **recurringPopModel** tag.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-18 recurringPopModel

Field Name	Description
priceTier	Defines recurring product offer price model. See "priceTier (RecurringPriceTier)".

priceTier (RecurringPriceTier)

Table 33-19 describes the fields in the **priceTier** (**RecurringPriceTier**) tag.

Table 33-19 priceTier (RecurringPriceTier)

Field Name	Description
lowerBound	Starting value for the range.
tierBasis	Container object to define charge tier expression.
enforceCreditLimit	Whether to enforce credit limits or let customers exceed them. See "Enforcing or Exceeding Credit Limits" for information and possible values.
tierRange	Container object to define charge tier range.

recurringCharge

Table 33-20 describes the fields in the **recurringCharge** tag.

Table 33-20 recurringCharge

Field Name	Description
price	The amount of debit or credit.
unitOfMeasure	The type of unit, such as Minute or Byte, to which the charge applies.
balanceElementNumCode	Numeric code for the balance element.
discountable	Boolean flag to make a balance impact discountable.
priceType	Type of impact: **grant** or **consumption**.
impactType	Used to specify whether the impact is fixed or scaled.
proratable	Boolean flag to set proration.

usageChargePopModel

Table 33-21 describes the fields in the **usageChargePopModel** tag.

Table 33-21 usageChargePopModel

Field Name	Description
priceTier	Defines usage charge product offer price model. See "priceTier".

priceTier

Table 33-22 describes the fields in the **priceTier** tag.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-22 priceTier

Field Name	Description
distributionMethod	Quantity range selection: • **UP_FRONT**: Pick the quantity range containing the value. • **FROM_BAL_IMPACT**: Distribute value across applicable quantity ranges.
tierBasis	Container object to define charge tier expression.
priceTierValidityPeriod	Validity period for the price tier. See "priceTierValidityPeriod".

priceTierValidityPeriod

Table 33-23 describes the fields in the **priceTierValidityPeriod** tag.

Table 33-23 priceTierValidityPeriod

Field Name	Description
lowerBound	Starting value for the range.
validFrom	Defines the start of the validity period. **0** means immediate.
priceTierRange	Container object to define price tier range. See "priceTierRange".

priceTierRange

Table 33-24 describes the fields in the **priceTierRange** tag.

Table 33-24 priceTierRange

Field Name	Description
upperBound	End value for the range. Supported for backward compatibility.
upperBoundExpression	Defines the end value for a quantity range or the noncurrency balance element to consume. See "upperBoundExpression".
scaledCharge	Defines scaled charge pop. See "scaledCharge".
fixedCharge	Defines fixed charge pop.

upperBoundExpression

Table 33-25 describes the fields in the **upperBoundExpression** tag.

Table 33-25 upperBoundExpression

Field Name	Description
balanceTBExpression	Defines the granted allowance to consume. See "balanceTBExpression".
numberTBExpression	Defines the end value for a quantity range. See "numberTBExpression".
autoRenewResourceExpression	Defines the automatic renewals of granted allowances. See "autoRenewResourceExpression".

Copyright © 2017, 2026, Oracle and/or its affiliates.

balanceTBExpression

Table 33-26 describes the fields in the **balanceTBExpression** tag.

Table 33-26 balanceTBExpression

Field Name	Description
balanceElementNumCode	The ID for the noncurrency balance element.

numberTBExpression

Table 33-27 describes the field in the **numberTBExpression** tag.

Table 33-27 numberTBExpression

Field Name	Description
value	The end value for a quantity range.

autoRenewResourceExpression

Table 33-28 describes the fields in the **autoRenewResourceExpression** tag.

Table 33-28 autoRenewResourceExpression

Field Name	Description
renewalLimitResource	Numeric code of a non-currency balance element that tracks the maximum number of renewals for a non-currency balance element.
renewalResource	Numeric code of the non-currency balance element to be renewed.
renewalAmountLimitResource	Numeric code of a non-currency balance element that tracks the maximum currency amount allowed for renewals.
maxCapResource	Numeric code of the non-currency balance element which defines the maximum limit after which renewals can happen.
maxCapResourceQuantity	Defines the quantity of the max cap resource.

scaledCharge

Table 33-29 describes the fields in the **scaledCharge** tag.

Table 33-29 scaledCharge

Field Name	Description
price	The amount to debit or credit.
unitOfMeasure	The type of unit, such as Minute or Byte, to which the charge applies.
balanceElementNumCode	Numeric code for the balance element, such as 640 for US dollars.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-29 (Cont.) scaledCharge

Field Name	Description
discountable	Boolean flag to make a balance impact discountable.
priceType	Type of impact: **GRANT** or **CONSUMPTION**.
incrementStep	The number of units to which the charge applies. For example, for a mobile phone service, you might charge 0.40 US dollars for every minute of usage in 2-minute increments. In this example: • Amount is 0.40. • Balance Element is US dollars. • Per Unit is Minute. • Increment is 2. **Note** BRM always prioritizes the **<incrementQuantity>** at the charge level over the **<IncrementStep>** at the pricing level. If both are configured in a charge using the PDC UI or the **ImportExportPricing** utility XML file, BRM selects the increment quantity.
incrementRounding	The type of rounding, which is based on the specified increment. For example, if a 3-minute-and-50-second call is rated in 2-minute increments, it is handled as follows: • **None**: Not applicable. (Applies only to an increment of 1). • **Round Down**: Charge for 2 minutes ($0.80). • **Round Up**: Charge for 4 minutes ($1.60).

Discount Offer

This section describes the XML tags that you use to configure discount offers in the XML pricing interface.

A discount offer is a purchasable product offering, similar to a charge offer. You can use discount offers to reduce a charge by a percentage, or to consume or grant balance elements. For more information, see "Creating Discount Offers".

alterationOffering

This is the main tag for configuring discount offers in PDC. Most of the tags are the same as for charge offers, except for the tags mentioned below. Table 33-30 describes the fields in the **alterationOffering** tag.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-30 alterationOffering

Field Name	Description
applicableChargeAndQuantity	Determines how the discounts are applied to the charge when a charge is eligible for multiple discount offers. • **ORIGINAL_CHARGE**: The discount offer is applied to the original charge amount, regardless of whether that amount was reduced by previous discount offers. • **REMAINING_CHARGE**: The discount offer is applied to the remaining charge amount after previous discount offers are applied. • **UNUSED_CHARGE_QUANTITY**: The discount offer is applied only to the part of the charge and the quantity that has not been used as the basis for a previous discount offer. This option is not valid for discount offers that credit a balance.
alterationEventMap	This is the XML tag inside the **AlterationOffering** (DO) XML tag which illustrates the event details related with the DO. This tag can occur multiple times in the DO tag based on how many events are mapped inside a Single DO. **Note**: You can map multiple events inside the DO but for override capability, you must only map single type of event under each DO. There will be one-to-one mapping of DO with **alterationEventMap**.
groupSharingEnabled	Boolean flag to enable the discount offer to be shared with all members of an automated discount sharing group when it is purchased by the group's owner. See "Sharing Discount Offers Automatically with Sharing Group Members" for information and examples.
tieToChargeOfferEnabled	Specifies whether the discount offer can be tied to specific charge offers at purchase time. Possible values are: • **true:** The discount offer is eligible for tying with specific charge offers. • **false:** The discount offer is not eligible for tying with specific charge offers. This is the default.
validityRounding	Specifies whether to start the discount offer's validity period at the purchase time or midnight of the purchase day. • **OFF**: Starts at the time of purchase. This overrides the CM **pin.conf** setting at the discount offer level. • **ON**: Starts at midnight (00:00:00) of the day the discount offer is purchased. This overrides the CM **pin.conf** setting at the discount offer level. • **NOT_SET**: Uses the systemwide setting in the CM **pin.conf** file. This is the default.

Copyright © 2017, 2026, Oracle and/or its affiliates.

AlterationEventMap

This is the mapping between event and alterationRatePlan (discount). Table 33-31 describes the fields in AlterationEventMap tag.

Table 33-31 AlterationEventMap

Field Name	Description
snowball	Displays if this discount is a snowball discount. Possible values are: • **true**: enable • **false**: disable In BRM, a snowball discount is a type of shared billing-time discount that distributes a discount to all accounts in a discount sharing group.
validAtStartNotValidAtEnd	How to apply discounts when they are valid from the start of a cycle but end in mid-cycle. Possible values are: • **FULL_DISCOUNT**: The discount is applied to the full cycle. • **PRORATE_DISCOUNT**: The discount is prorated. • **NO_DISCOUNT**: The discount is *not* applied. • **NOT_APPLICABLE**: The discount is not applicable.
notValidAtStartValidAtEnd	How to apply discounts when they are valid from mid-cycle to the end of the cycle. Possible values are: • **FULL_DISCOUNT**: The discount is applied to the full cycle. • **PRORATE_DISCOUNT**: The discount is prorated. • **NO_DISCOUNT**: The discount is *not* applied. • **NOT_APPLICABLE**: The discount is not applicable.
notValidAtStartNotValidAtEnd	How to apply discounts when they start mid-cycle and end mid-cycle. Possible values are: • **FULL_DISCOUNT**: The discount is applied to the full cycle. • **PRORATE_DISCOUNT**: The discount is prorated. • **NO_DISCOUNT**: The discount is *not* applied. • **NOT_APPLICABLE**: The discount is not applicable.
alterationRatePlanName	This refers to the name of the discount configuration inside the DO at a granular and more detailed level. Discount Configuration name can be any name such as OneTime discount, Recurring Discount, RatingTimeDiscount, and so on.
ratePlanIID	Discount rate plan IID generated internally by PDC system for the **alterationEventMap** XML tag.

Copyright © 2017, 2026, Oracle and/or its affiliates.

alterationRatePlan

Root tag for the alteration rate plan configuration. Table 33-32 describes the fields in the **alterationRatePlan** tag.

For an example rate plan configuration, see "Consuming Noncurrency Resources for Discounts in Order of Expiration".

Table 33-32 alterationRatePlan

Field Name	Description
internalId	PDC system generated internal id for the objects.
pricingProfileName	Standard Discount for usage event. Subscription Discount for onetime and recurring events.
priceListName	This is configured as **Default** by default.
obsolete	Boolean flag to obsolete the promoted objects.
taxCode	The tax code used for the discount. You can use either this parameter or **TaxSelectorGroup**, but not both.
TaxSelectorGroup	The name of the tax selector group to use for the discount. You can use either this parameter or **taxCode**, but not both.
resourcesConsumptionOrder	Whether to consume noncurrency resources for the discounts in order of earliest expiry, or in order of any discount rules configured: • **EARLIEST_EXPIRY**: Consume noncurrency resources according to which resource expires first. For example, if the customer is subscribed to one discount that grants minutes that expire monthly and another that grants minutes that expire yearly, the minutes expiring monthly should be consumed first. If all resources have the same validity dates, the discount rule order will be used. • **RULE_ORDER**: Consume balances according to the configured discount rules, without taking validity into account.
arpDateRange	Alteration rate plan relative date range.
QuantityTriggerExpression	An expression that handles quantities such as data usage or call minutes, and calculates pricing or discounts based on these usage amounts.
BalanceTriggerExpression	An expression that refers to pricing and discount logic based on available balances.
ChargeTriggerExpression	An expression that represents charge-related calculations, where the discount is determined based on the base charge calculated for the rate plan.
CustomTriggerExpression	An expression that defines custom calculations or logic not covered by standard expressions.
UnaryTriggerExpression	Specifies a trigger condition based on recursive unary expression. It evaluates one input using operators to compare it to constants.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-32 (Cont.) alterationRatePlan

Field Name	Description
ComplexTriggerExpression	Specifies a trigger condition based on recursive binary expression. Contains **binaryExpression** and additional trigger attributes: operator and value.

QuantityTriggerExpression

Table 33-33 QuantityTriggerExpression

Field Name	Description
Operator	Specifies how to compare an expression to a value, using **Equal To, Greater Than, Greater Than or Equal To, Less Than, Less Than or Equal To,** or **Not Equal To**.
Value	A numerical value that the QuantityTriggerExpression is compared with.

BalanceTriggerExpression

Table 33-34 BalanceTriggerExpression

Field Name	Description
Operator	Specifies how to compare an expression to a value, using **Equal To, Greater Than, Greater Than or Equal To, Less Than, Less Than or Equal To,** or **Not Equal To**.
Value	A numerical value that the BalanceTriggerExpression is compared with.
balanceElementNumCode	The ID of the balance element configured.
memberBalance	A Boolean flag in the balance expression that indicates to ECE that the member balance should be used instead of the owner balance when evaluating the expression.
includeReservations	Optional. Specifies whether ECE includes active reservations in the balance amount used to determine whether a trigger condition is met. Valid values are **true** and **false**. When set to **true**, ECE includes amounts reserved for ongoing sessions. When set to **false**, ECE ignores active reservations and checks the balance without the reserved amounts. If this value is not specified, ECE uses **true**.

Copyright © 2017, 2026, Oracle and/or its affiliates.

ChargeTriggerExpression

Table 33-35 ChargeTriggerExpression

Field Name	Description
Operator	Specifies how to compare an expression to a value, using **Equal To, Greater Than, Greater Than or Equal To, Less Than, Less Than or Equal To,** or **Not Equal To**.
Value	A numerical value that the ChargeTriggerExpression is compared with.

CustomTriggerExpression

Table 33-36 CustomTriggerExpression

Field Name	Description
Operator	Specifies how to compare an expression to a value, using **Equal To, Greater Than, Greater Than or Equal To, Less Than, Less Than or Equal To,** or **Not Equal To**.
Value	A numerical value that the CustomTriggerExpression is compared with.
functionName	Accepts a user-defined name.

UnaryTriggerExpression

Table 33-37 UnaryTriggerExpression

Field Name	Description
Operator	Specifies how to compare an expression to a value, using **Equal To, Greater Than, Greater Than or Equal To, Less Than, Less Than or Equal To,** or **Not Equal To**.
Value	A numerical value that the UnaryTriggerExpression is compared with.
unaryExpression	Contains the **Square Root** or **Decimal Absolute** expressions, and uses a single operand to evaluate pricing details.
Argument	A subtag within unaryExpression that holds the expression's single operand. This operand can be any other expression type, **binary**, **balance**, **number**, **charge**, **quantity**, **custom**, or another unary. For example, if the operand is a binary expression, the argument tag contains the entire binaryExpression.
leftOperand	Expression to use as the left side of the computation.
numberExpression	Defines the scalar value for arithmetic operations.
rightOperand	Expression to use as the right side of the computation.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-37 (Cont.) UnaryTriggerExpression

Field Name	Description
Number	Specifies the numeric value of **numberExpression** tag.
binaryOperator	Uses binary operators such as **+**, **-**, *****, **/**, **Max**, **Min**, and **Round**.
functionName	Accepts the expressions **sqrt** and **decimalAbs**.

ComplexTriggerExpression

Table 33-38 ComplexTriggerExpression

Field Name	Description
Operator	Specifies how to compare an expression to a value, using **Equal To, Greater Than, Greater Than or Equal To, Less Than, Less Than or Equal To,** or **Not Equal To**.
Value	The numeric value that ComplexTriggerExpression is compared with.
binaryExpression	Sub-entry that includes the binaryExpression.
leftOperand	Uses an expression used as the left side of the computation.
numberExpression	Defines the scalar value for arithmetic operations.
Number	Specifies the numeric value of the **numberExpression** tag.
rightOperand	Uses an expression used as the right side of the computation.
binaryOperator	Uses binary operators such as **+**, **-**, *****, **/**, **Max**, **Min**, and **Round**.

TaxSelectorGroup

The **TaxSelectorGroup** tag represents the tax selector information for the discount. Table 33-39 describes the fields in the **TaxSelectorGroup** tag.

Table 33-39 TaxExemptionSelectorGroup

Field Name	Description
taxSelectorName	The name of the tax selector
taxSelectorIID	The ID of the tax selector

arpDateRange

Table 33-40 describes the fields in the **arpDateRange** tag.

Table 33-40 arpDateRange

Field Name	Description
startDate	Start date for a date range in a discount.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-40 (Cont.) arpDateRange

Field Name	Description
endDate	End date for a date range in a discount.
alterationConfiguration	Defines configuration details for the alteration.

alterationConfiguration

Table 33-41 describes the fields in the **alterationConfiguration** tag.

Table 33-41 alterationConfiguration

Field Name	Description
applicableChargeAndQuantity	Select how to apply the discount. By default, a discount rule applies the discounts in the same way as the discount offer that contains the rule. • **Original Charge**: The discount offer is applied to the original charge amount, regardless of whether that amount was reduced by previous discount offers. • **Remaining Charge**: The discount offer is applied to the remaining charge amount after previous discount offers are applied. • **Remaining Charge and Quantity**: The discount offer is applied only to the part of the charge and quantity that has not been used as the basis for a previous discount offer.
arpCompositePopModel	Alteration rate plan composite pop model. Holds information about the pricing/balImpact used in alteration offer pricing.
triggerSpec	Defines the trigger's behavior for calculating the Absolute Function when data is inserted, updated, or deleted.

arpCompositePopModel

Table 33-42 describes the fields in the **arpCompositePopModel** tag.

Table 33-42 arpCompositePopModel

Field Name	Description
alterationPopModel	Defines alteration product offer price model.

alterationPopModel

Table 33-43 describes the fields in the **alterationPopModel** tag.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-43 alterationPopModel

Field Name	Description
priceTier	If a pricing uses more than one RUM, a tab appears for each RUM. These tabs are called price tiers.

priceTier

Table 33-44 describes the fields in the **priceTier** tag.

Table 33-44 priceTier

Field Name	Description
distributionMethod	Quantity range selection: • **UP_FRONT**: Pick the quantity range containing the value. • **FROM_BAL_IMPACT**: Distribute value across applicable quantity ranges.
tierBasis	Container object to define charge tier expression.
lowerBound	Starting value for the range. Expressions are supported.
tierRange	Container object to define charge tier range.

tierBasis

Table 33-45 describes the fields in the **tierBasis** tag.

Table 33-45 tierBasis

Field Name	Description
chargeTierExpression	Expression for the charge tier.
unaryTierExpression	The tierBasis expression for Unary operation. It configures the functions decimalAbs() and sqrt() for alterationRatePlan and distributionRatePlan.
customTierExpression	An expression defined by the user.
binaryTierExpression	The tierBasis expression for Binary operation.
numberTierExpression	The number expression for tierBasis.
quantityTierExpression	The tierBasis expression for RUM Quantity.
balanceTierExpression	The tierBasis expression for Balance.

chargeTierExpression

Table 33-46 describes the fields in the **chargeTierExpression** tag.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-46 chargeTierExpression

Field Name	Description
useTierComponent	Defines which of the tiers would qualify for impact. Conventionally, it maps to Threshold/Tiered discount. The amount is either in one quantityRange or distributed across several quantityRanges.
preRated	A Boolean flag to indicate that the charge is prerated.

UnaryTierExpression

Table 33-47 describes the fields in the **UnaryTierExpression** tag.

Table 33-47 UnaryTierExpression

Field Name	Description
functionName	Accepts the expressions **sqrt** and **decimalAbs**.
argument	A subtag within unaryExpression that holds the expression's single operand. This operand can be any other expression type, **binary**, **balance**, **number**, **charge**, **quantity**, **custom**, or another unary. For example, if the operand is a binary expression, the argument tag contains the entire binaryExpression.
balanceBasisExpression	An expression that refers to pricing and discount logic based on available balances.
balanceElementNumCode	The ID of balance element configured.
memberBalance	A Boolean flag in the balance expressions that indicates to ECE that the member balance should be used instead of the owner balance when evaluating the expression.

customTierExpression

Table 33-48 describes the fields in the **customTierExpression** tag.

Table 33-48 customTierExpression

Field Name	Description
functionName	Accepts a user-defined name.

BinaryTierExpression

Table 33-57 describes the fields in the **BinaryTierExpression** tag.

Table 33-49 BinaryTierExpression

Field Name	Description
binaryOperator	Uses binary operators such as **+**, **-**, *****, **/**, **Max**, **Min**, and **Round**.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-49 (Cont.) BinaryTierExpression

Field Name	Description
leftOperand	Uses an expression used as the left side of the computation.
rightOperand	Uses an expression used as the right side of the computation.
quantityBasisExpression	An expression that handles quantities such as data usage or call minutes, and calculates pricing or discounts based on these usage amounts.
useTierComponent	Defines which of the tiers would qualify for impact. Conventionally, it maps to Threshold/Tiered discount. The amount is either in one quantityRange or distributed across several quantityRanges.
numberBasisExpression	An expression which represents numeric expressions and calculates monetary amounts.
value	The numerical value of numberBasisExpression.

QuantityTierExpression

Table 33-56 describes the fields in the **QuantityTierExpression** tag.

Table 33-50 QuantityTierExpression

Field Name	Description
useTierComponent	Defines which of the tiers would qualify for impact. Conventionally, it maps to Threshold/Tiered discount. The amount is either in one quantityRange or distributed across several quantityRanges.

BalanceTierExpression

Table 33-58 describes the fields in the **BalanceTierExpression** tag.

Table 33-51 BalanceTierExpression

Field Name	Description
balanceElementNumCode	The ID of balance element configured.
memberBalance	A Boolean flag in the balance expressions that indicates to ECE that the member balance should be used instead of the owner balance when evaluating the expression.

tierRange

Table 33-52 describes the fields in the **tierRange** tag.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-52 tierRange

Field Name	Description
upperBound	Maximum value of the range. Expressions are supported.
percentAlteration	Defines percentage Alteration value.
fixedAlteration	Defines fixed Alteration value.

percentAlteration

Table 33-53 describes the fields in the **percentAlteration** tag.

Table 33-53 percentAlteration

Field Name	Description
price	It is applied towards the amount that is discounted.
unitOfMeasure	The type of unit, such as Minute or Byte, to which the discount applies.
balanceElementNumCode	Select the currency or noncurrency balance element to debit or credit.
alterationAppliesTo	Select the balance to apply the discount to. • **User Balance**: The balance of the account or service that generates the event. • **Sharer Balance**: The balance of the account or service that owns the discount.
alterationBasedOn	Defines how much alteration/discount should be applied. It is defined at the pricing level.
priceType	Type of impact: **grant** or **consumption**.
taxCode	The tax code used for the discount.
glid	Used for selecting the required GL Information for a given balance impact.
priceValidity	Defines validity for the price.

alterationBasedOn

Table 33-54 describes the fields in the **alterationBasedOn** tag.

Table 33-54 alterationBasedOn

Field Name	Description
expression	An expression which represents the **What to Discount** field in the UI.
NumberBasisExpression	An expression which represents numeric expressions and calculates monetary amounts.
QuantityBasisExpression	An expression that handles quantities such as data usage or call minutes, and calculates pricing or discounts based on these usage amounts.
BinaryBasisExpression	An expression that evaluates two inputs using binary operators such as +, -, /, Max, Min, or Round.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-54 (Cont.) alterationBasedOn

Field Name	Description
BalanceBasisExpression	An expression that refers to pricing and discount logic based on available balances.
ChargeBasisExpression	Defines an expression for various fields such as **QuantityRange**, **DiscountTrigger**, **tierBasis** (DRUM), and **BalanceImpact** (Pricing).
CustomBasisExpression	The tier basis expression defined by the user. Defines custom calculations or logic not covered by standard expressions.
UnaryBasisExpression	An expression that defines Unary operations with exactly one operand, which can be another expression.

NumberBasisExpression

Table 33-55 describes the fields in the **NumberBasisExpression** tag.

Table 33-55 NumberBasisExpression

Field Name	Description
Value	The numerical value of numberBasisExpression.

QuantityBasisExpression

Table 33-56 describes the fields in the **QuantityBasisExpression** tag.

Table 33-56 QuantityBasisExpression

Field Name	Description
useTierComponent	Defines which of the tiers would qualify for impact. Conventionally, it maps to Threshold/Tiered discount. The amount is either in one quantityRange or distributed across several quantityRanges.

BinaryBasisExpression

Table 33-57 describes the fields in the **BinaryBasisExpression** tag.

Table 33-57 BinaryBasisExpression

Field Name	Description
binaryOperator	Uses binary operators such as **+**, **-**, *****, **/**, **Max**, **Min**, and **Round**.
leftOperand	Uses an expression used as the left side of the computation.
rightOperand	Uses an expression used as the right side of the computation.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-57 (Cont.) BinaryBasisExpression

Field Name	Description
quantityBasisExpression	An expression that handles quantities such as data usage or call minutes, and calculates pricing or discounts based on these usage amounts.
useTierComponent	Defines which of the tiers would qualify for impact. Conventionally, it maps to Threshold/Tiered discount. The amount is either in one quantityRange or distributed across several quantityRanges.
numberBasisExpression	An expression which represents numeric expressions and calculates monetary amounts.
value	The numerical value of numberBasisExpression.

BalanceBasisExpression

Table 33-58 describes the fields in the **BalanceBasisExpression** tag.

Table 33-58 BalanceBasisExpression

Field Name	Description
balanceElementNumCode	The ID of balance element configured.
memberBalance	A Boolean flag in the balance expressions that indicates to ECE that the member balance should be used instead of the owner balance when evaluating the expression.

ChargeBasisExpression

Table 33-59 describes the fields in the **ChargeBasisExpression** tag.

Table 33-59 ChargeBasisExpression

Field Name	Description
useTierComponent	Defines which of the tiers would qualify for impact. Conventionally, it maps to Threshold/Tiered discount. The amount is either in one quantityRange or distributed across several quantityRanges.
preRated	A Boolean flag to indicate that the charge is prerated.

CustomBasisExpression

Table 33-60 describes the fields in the **CustomBasisExpression** tag.

Table 33-60 CustomBasisExpression

Field Name	Description
functionName	Accepts a user-defined name.

Copyright © 2017, 2026, Oracle and/or its affiliates.

UnaryBasisExpression

Table 33-61 describes the fields in the **UnaryBasisExpression** tag.

Table 33-61 UnaryBasisExpression

Field Name	Description
functionName	Accepts the expressions **sqrt** and **decimalAbs**.
argument	A subtag within unaryExpression that holds the expression's single operand. This operand can be any other expression type, **binary**, **balance**, **number**, **charge**, **quantity**, **custom**, or another unary. For example, if the operand is a binary expression, the argument tag contains the entire binaryExpression.
balanceBasisExpression	An expression that refers to pricing and discount logic based on available balances.
balanceElementNumCode	The ID of balance element configured.
memberBalance	A Boolean flag in the balance expressions that indicates to ECE that the member balance should be used instead of the owner balance when evaluating the expression.

priceValidity

Table 33-62 describes the fields in the **priceValidity** tag.

Table 33-62 priceValidity

Field Name	Description
priceValidity	To define the validity period for the credited amount in the balance impact.
startValidityMode	Defines start validity mode: • **FOREVER** • **IMMEDIATE** • **ABSOLUTE** • **RELATIVE** • **FIRST_USAGE** • **PERIOD_START**
endValidityMode	Defines end validity mode: • **NEVER** • **ABSOLUTE** • **RELATIVE_TO_START** • **RELATIVE_TO_PERIOD_START**
validityRange	Defines the validity range. **DateTimeRange** should be defined as *YYYYMMDDTHHMMSS* or *YYYYMMDDTHHMMSS*.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-62 (Cont.) priceValidity

Field Name	Description
relativeStartOffset	To make the amount valid in any number of accounting cycles, billing cycles, months, days, hours, minutes, or seconds from the event occurrence. For example, if you specify two accounting cycles, the amount will be valid after two accounting cycles from when the event occurs.
relativeEndOffset	The amount validity ends relative to the validity start time. Enter the number of accounting cycles, billing cycles, months, days, calendar days, hours, minutes, or seconds from the validity start time when the validity ends.
relativeStartOffsetUnit	Accounting cycles, billing cycles, months, days, hours, minutes, or seconds.
relativeEndOffsetUnit	Accounting cycles, billing cycles, months, days, calendar days, hours, minutes, or seconds.

fixedAlteration

Table 33-63 describes the fields in the **fixedAlteration** tag.

Table 33-63 fixedAlteration

Field Name	Description
price	It is applied towards the amount that is discounted. It must be a positive value.
unitOfMeasure	The type of unit, such as Minute or Byte, to which the discount applies.
balanceElementNumCode	Select the currency or noncurrency balance element to debit or credit.
alterationAppliesTo	Select the balance to apply the discount to: • **User Balance**: The balance of the account or service that generates the event. • **Sharer Balance**: The balance of the account or service that owns the discount.
alterationBasedOn	Defines how much alteration/discount should be applied. It is defined at the pricing level.
priceType	Type of impact: **grant** or **consumption**.
taxCode	The tax code used for the discount.
glid	Used for selecting the required GL information for a given balance impact.
priceValidity	Defines validity for the price.

Chargeshare Offer

This section describes the XML tags that you use to configure chargeshares in the XML pricing interface.

Copyright © 2017, 2026, Oracle and/or its affiliates.

A chargeshare offer contains one or more chargeshares that specify how charges are shared among the members of a group. Charge sharing enables an account to sponsor the charges of other accounts in the group. The sponsoring account receives the balance impact of charges incurred by the group member accounts. For more information, see "Sharing Charges and Discounts".

distributionOffering

Table 33-64 describes the fields in the **distributionOffering** tag.

Table 33-64 distributionOffering

Field Name	Description
name	Unique name for the chargeshare offer.
description	Description for the chargeshare offer.
internalId	Unique id internally generated by PDC while committing the pricing configuration. It cannot be same for any other pricing or configuration entity.
pricingProfileName	Pricing profile to associate with the chargeshare offer.
priceListName	This is configured as **Default** by default.
obsolete	Boolean flag to obsolete the promoted (or active) objects.
timeRange	The time range during which the offer is available for purchase. Format: starttime/endtime **Examples**: • <timeRange>**0/inf**</timeRange> indicates the offer is available for purchase immediately and there is no end date. • <timeRange>**20220712T001137/20220730T001144**</timeRange> indicates the offer can be purchased only within the time period mentioned.
productSpecName	The service type to which the charge offer applies.
offerType	Select Use for Aggregation. The ChargeShare offer is set for aggregation.

distributionEventMap

This is the XML tag inside the **distributionOffering** (charge offer) XML tag which illustrates the event details related with the chargeshare offer. This tag can occur multiple times in the discount offer tag based on how many events are mapped inside a single discount offer.

Table 33-65 describes the fields in the **distributionEventMap** tag.

> **Note**
>
> You can map multiple events inside the discount offer, but for override capability, only map single type of event under each discount offer. There will be one-to-one mapping of discount offer with **distributionEventMap**.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-65 distributionEventMap

Field Name	Description
eventName	The name of the event. For example, EventDelayedSessionTelcoGsm (**/event/delayed/session/telco/gsm** in BRM) would be used for usage events, while EventBillingProductFeeCycleCycle_forward_monthly (**/event/billing/product/fee/cycle/cycle_forward_monthly**) would represent a monthly recurring fee (in-advance).
validIfCancelled	Charge even if the instantiated product offering status is canceled. BRM offering statuses are: • **1** = active • **2** = inactive • **3** = canceled
validIfInactive	Charge even if the instantiated product offering status is inactive.
timezoneMode	Determines the time zone which will be used to rate an event: • From ACCOUNT: Uses the time zone in which the account is registered. • From EVENT: Uses the time zone in which the event occurs. For example, if you make a call from California and your account is registered in New York: • If you select From Account, New York time is used. • If you select From Event, California time is used.
minQuantity	Minimum value that the RUM quantity is rounded to. Value smaller than min_quantity is rounded to min_quantity. Minimum quantity starts from 0 by default. For example, if an event had a duration of 10 seconds, and the minimum quantity was configured as 30, then 30 seconds would be rated. If the duration of the event were 45 seconds, 45 seconds would be rated (as it is larger than the minimum configured).
minQuantityUnit	Unit in which min_quantity is expressed. Possible values are: • NONE (**0**) • SECOND (**1**) • MINUTE (**2**) • HOUR (**3**) • DAY (**4**)
incrementQuantity	Incremental value to round RUM Quantity to. RUM Quantity is always rounded to an integral multiple of the Incremental Quantity. For example, if the incrementQuantity was configured as 30 for a RUM of seconds, then the amount to rate would be rounded to that increment (see "roundingMode") and a 43 second call would be rated as 60 seconds (if rounding up) or 30 seconds (if rounding down).

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-65 (Cont.) distributionEventMap

Field Name	Description
incrementQuantityUnit	Unit associated with the increment quantity. For example, for an increment of 60 seconds, you would enter Seconds.
roundingMode	How to round the RUM quantity. Possible values are: • Up (**0**) • Down (**1**) • Nearest (**2**)
prorateFirst	Proration setting when purchased mid-cycle. It can be either **FULL_CHARGE** or **PRORATE_CHARGE**.
prorateLast	The proration setting when canceled mid-cycle. It can be either **FULL_CHARGE** or **PRORATE_CHARGE**.
distributionRatePlanName	The name of the charge configuration inside the discount offer at a granular level. Enter any name, such as One-time charge, MRC, Rollover, FreeSMS, FreeVoice, and so on.
ratePlanIID	Charge rate plan IID generated internally by the PDC system for the **distributionEventMap** XML tag.
distributionRatePlanInfo	The target engine where the pricing or configuration information is published. • **ECE** = ECE rating engine • **BRE** = Batch rating synchronized to Pipeline Manager • **RRE** = Real-time rating synchronized to BRM

distributionRatePlanInfo

Table 33-66 describes the fields in the **distributionRatePlanInfo** tag.

Table 33-66 distributionRatePlanInfo

Field Name	Description
targetEngine	The target engine where to publish information about the discount offer. Possible values are: • **BRE** • **ECE** • **RRE** This is automatically taken care by the **ImportExportPricing** utility in PDC as per the charge type and charge category.

distributionRatePlan

Root tag for the distribution rate plan configuration. Table 33-67 describes the fields in the **distributionRatePlan** tag.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-67 distributionRatePlan

Field Name	Description
internalId	PDC system-generated internal ID for the objects.
pricingProfileName	**Convergent Usage** for usage events. **Subscription** for one-time and recurring events.
priceListName	This is configured as **Default**.
obsolete	Boolean flag to obsolete the promoted objects.
taxCode	The tax code used for the chargeshare. You can use either this parameter or TaxSelectorGroup, but not both.
TaxSelectorGroup	The name of the tax selector group to use for the chargeshare. You can use either this parameter or taxCode, but not both.
drpDateRange	Distribution rate plan relative date range.

TaxSelectorGroup

The **TaxSelectorGroup** tag represents the tax selector information for the chargeshare. Table 33-68 describes the fields in the **TaxSelectorGroup** tag.

Table 33-68 TaxExemptionSelectorGroup

Field Name	Description
taxSelectorName	The name of the tax selector
taxSelectorIID	The ID of the tax selector

drpDateRange

Table 33-69 describes the fields in the **drpDateRange** tag.

Table 33-69 drpDateRange

Field Name	Description
startDate	Start date for a date range in a charge share.
endDate	End date for a date range in a charge share.
distributionConfiguration	Container object to define configurations for the distribution rate plan.

distributionConfiguration

Table 33-70 describes the fields in the **distributionConfiguration** tag.

Table 33-70 distributionConfiguration

Field Name	Description
drpCompositePopModel	Distribution rate plan composite pop model. Holds information about the pricing/balImpact used in the distribution offer pricing.

Copyright © 2017, 2026, Oracle and/or its affiliates.

drpCompositePopModel

Table 33-71 describes the fields in the **drpCompositePopModel** tag.

Table 33-71 drpCompositePopModel

Field Name	Description
alterationPopModel	An alteration pop model rate plan element which contains the price tier information.

alterationPopModel

Table 33-72 describes the fields in the **alterationPopModel** tag.

Table 33-72 alterationPopModel

Field Name	Description
priceTier	If a pricing uses more than one RUM, enter the price tier name.

priceTier

Table 33-73 describes the fields in the **priceTier** tag.

Table 33-73 priceTier

Field Name	Description
distributionMethod	Quantity range selection: • **UP_FRONT**: The quantity range containing the value. • **FROM_BAL_IMPACT**: Distribute the value across applicable quantity ranges.
tierBasis	Container object to define charge tier expression.
lowerBound	Starting value for the range. Expressions are supported.
tierRange	Container object to define charge tier range.

tierBasis

Table 33-74 describes the fields in the **tierBasis** tag.

Table 33-74 tierBasis

Field Name	Description
chargeTierExpression	Expression for the charge tier.
unaryTierExpression	The tierBasis expression for Unary operation. It configures the functions decimalAbs() and sqrt()) for alterationRatePlan and distributionRatePlan.
customTierExpression	An expression defined by the user.
binaryTierExpression	The tierBasis expression for Binary operation.
numberTierExpression	The number expression for tierBasis.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-74 (Cont.) tierBasis

Field Name	Description
quantityTierExpression	The tierBasis expression for RUM Quantity.
balanceTierExpression	The tierBasis expression for Balance.

chargeTierExpression

Table 33-75 describes the fields in the **chargeTierExpression** tag.

Table 33-75 chargeTierExpression

Field Name	Description
useTierComponent	Defines which of the tiers would qualify for impact. Conventionally, it maps to Threshold/Tiered discount. The amount is either in one quantityRange or distributed across several quantityRanges.
preRated	A Boolean flag to indicate that the charge is prerated.

UnaryTierExpression

Table 33-76 describes the fields in the **UnaryTierExpression** tag.

Table 33-76 UnaryTierExpression

Field Name	Description
functionName	Accepts the expressions **sqrt** and **decimalAbs**.
argument	A subtag within unaryExpression that holds the expression's single operand. This operand can be any other expression type, **binary**, **balance**, **number**, **charge**, **quantity**, **custom**, or another unary. For example, if the operand is a binary expression, the argument tag contains the entire binaryExpression.
balanceBasisExpression	An expression that refers to pricing and discount logic based on available balances.
balanceElementNumCode	The ID of the balance element configured.
memberBalance	A Boolean flag in the balance expression that indicates to ECE that the member balance should be used instead of the owner balance when evaluating the expression.

customTierExpression

Table 33-77 describes the fields in the **customTierExpression** tag.

Table 33-77 customTierExpression

Field Name	Description
functionName	Accepts a user-defined name.

Copyright © 2017, 2026, Oracle and/or its affiliates.

BinaryTierExpression

Table 33-78 describes the fields in the **BinaryTierExpression** tag.

Table 33-78 BinaryTierExpression

Field Name	Description
binaryOperator	Uses binary operators such as **+**, **-**, *****, **/**, **Max**, **Min**, and **Round**.
leftOperand	Uses an expression used as the left side of the computation.
rightOperand	Uses an expression used as the right side of the computation.
quantityBasisExpression	An expression that handles quantities such as data usage or call minutes, and calculates pricing or discounts based on these usage amounts.
useTierComponent	Defines which of the tiers would qualify for impact. Conventionally, it maps to Threshold/Tiered discount. The amount is either in one quantityRange or distributed across several quantityRanges.
numberBasisExpression	An expression which represents numeric expressions and calculates monetary amounts.
value	The numerical value of numberBasisExpression.

QuantityTierExpression

Table 33-79 describes the fields in the **QuantityTierExpression** tag.

Table 33-79 QuantityTierExpression

Field Name	Description
useTierComponent	Defines which of the tiers would qualify for impact. Conventionally, it maps to Threshold/Tiered discount. The amount is either in one quantityRange or distributed across several quantityRanges.

BalanceTierExpression

Table 33-80 describes the fields in the **BalanceTierExpression** tag.

Table 33-80 BalanceTierExpression

Field Name	Description
balanceElementNumCode	The ID of balance element configured.
memberBalance	A Boolean flag in the balance expressions that indicates to ECE that the member balance should be used instead of the owner balance when evaluating the expression.

Copyright © 2017, 2026, Oracle and/or its affiliates.

lowerBound

Table 33-81 describes the fields in the **lowerBound** tag.

Table 33-81 lowerBound

Field Name	Description
numberTBExpression	The number expression for charge tier bound.

numberTBExpression

Table 33-82 describes the fields in the numberTBExpression tag.

Table 33-82 numberTBExpression

Field Name	Description
value	Numeric value for Tier Bound. • Float.POSITIVE_INFINITY for Inf • Float.NEGATIVE_INFINITY for -Inf

tierRange

Table 33-83 describes the fields in the **tierRange** tag.

Table 33-83 tierRange

Field Name	Description
upperBound	Maximum value of the range. Expressions are supported.
percentAlteration	Defines percentage discount value.

upperBound

Table 33-84 describes the fields in the **upperBound** tag.

Table 33-84 upperBound

Field Name	Description
numberTBExpression	The number expression for charge tier bound.

numberTBExpression

Table 33-85 describes the fields in the **numberTBExpression** tag.

Table 33-85 numberTBExpression

Field Name	Description
value	Numeric value for tier bound: • Float.POSITIVE_INFINITY for Inf • Float.NEGATIVE_INFINITY for -Inf

Copyright © 2017, 2026, Oracle and/or its affiliates.

percentAlteration

Table 33-86 describes the fields in the **percentAlteration** tag.

Table 33-86 percentAlteration

Field Name	Description
price	The amount that is discounted.
unitOfMeasure	The type of unit, such as Minute or Byte, to which the discount applies.
balanceElementNumCode	The currency or noncurrency balance element to debit or credit.
alterationAppliesTo	The balance to apply the discount to: • **User Balance**: The balance of the account or service that generates the event. • **Sharer Balance**: The balance of the account or service that owns the discount.
alterationBasedOn	Defines how much the discount should be applied. It is defined at the pricing level.
priceType	Type of impact. Valid values are: • **GRANT** • **CONSUMPTION** • **AGGREGATE**
priceValidity	To define the validity period for the credited amount in the balance impact.

alterationBasedOn

Table 33-87 describes the fields in the **alterationBasedOn** tag.

Table 33-87 alterationBasedOn

Field Name	Description
chargeBasisExpression	Defines an expression for various fields such as **QuantityRange**, **DiscountTrigger**, **tierBasis** (DRUM), and **BalanceImpact** (Pricing).
NumberBasisExpression	An expression which represents numeric expressions and calculates monetary amounts.
QuantityBasisExpression	An expression that handles RUM quantities such as data usage or call minutes, and calculates pricing or discounts based on these usage amounts.
BinaryBasisExpression	An expression that evaluates two inputs using binary operators such as +, -, /, Max, Min, or Round.
BalanceBasisExpression	An expression that refers to pricing and discount logic based on available balances.
CustomBasisExpression	The tier basis expression defined by the user. It defines custom calculations or logic not covered by standard expressions.
UnaryBasisExpression	An expression that defines Unary operations with exactly one operand, which can be another expression.

Copyright © 2017, 2026, Oracle and/or its affiliates.

chargeBasisExpression

Table 33-88 describes the fields in the **chargeBasisExpression** tag.

Table 33-88 chargeBasisExpression

Field Name	Description
useTierComponent	Defines which tiers qualify for impact. Conventionally, it maps to Threshold/Tiered discount. The amount is either in one quantityRange or distributed across several quantityRanges.
preRated	A Boolean flag for indicating whether the charge is prerated.

NumberBasisExpression

Table 33-88 describes the fields in the **NumberBasisExpression** tag.

Table 33-89 NumberBasisExpression

Field Name	Description
Value	The numerical value of numberBasisExpression.

QuantityBasisExpression

Table 33-89 describes the fields in the **QuantityBasisExpression** tag.

Table 33-90 QuantityBasisExpression

Field Name	Description
useTierComponent	Defines which of the tiers would qualify for impact. Conventionally, it maps to Threshold/Tiered discount. The amount is either in one quantityRange or distributed across several quantityRanges.

BinaryBasisExpression

Table 33-91 describes the fields in the **BinaryBasisExpression** tag.

Table 33-91 BinaryBasisExpression

Field Name	Description
binaryOperator	Uses binary operators such as **+**, **-**, *****, **/**, **Max**, **Min**, and **Round**.
leftOperand	Expression to use as the left side of the computation.
rightOperand	Expression to use as the right side of the computation.
quantityBasisExpression	An expression that handles quantities such as data usage or call minutes, and calculates pricing or discounts based on these usage amounts.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-91 (Cont.) BinaryBasisExpression

Field Name	Description
useTierComponent	Defines which of the tiers would qualify for impact. Conventionally, it maps to Threshold/Tiered discount. The amount is either in one quantityRange or distributed across several quantityRanges.
numberBasisExpression	An expression which represents numeric expressions and calculates monetary amounts.
value	The numerical value of numberBasisExpression.

BalanceBasisExpression

Table 33-92 describes the fields in the **BalanceBasisExpression** tag.

Table 33-92 BalanceBasisExpression

Field Name	Description
balanceElementNumCode	The ID of balance element configured.
memberBalance	A Boolean flag in the balance expressions that indicates to ECE that the member balance should be used instead of the owner balance when evaluating the expression.

CustomBasisExpression

Table 33-93 describes the fields in the **CustomBasisExpression** tag.

Table 33-93 CustomBasisExpression

Field Name	Description
functionName	Accepts a user-defined name.

UnaryBasisExpression

Table 33-94 describes the fields in the **UnaryBasisExpression** tag.

Table 33-94 UnaryBasisExpression

Field Name	Description
functionName	Accepts the expressions **sqrt** and **decimalAbs**.
argument	A subtag within unaryExpression that holds the expression's single operand. This operand can be any other expression type, **binary**, **balance**, **number**, **charge**, **quantity**, **custom**, or another unary. For example, if the operand is a binary expression, the argument tag contains the entire binaryExpression.
balanceBasisExpression	An expression that refers to pricing and discount logic based on available balances.
balanceElementNumCode	The ID of balance element configured.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-94 (Cont.) UnaryBasisExpression

Field Name	Description
memberBalance	A Boolean flag in the balance expressions that indicates to ECE that the member balance should be used instead of the owner balance when evaluating the expression.

priceValidity

Table 33-95 describes the fields in the **priceValidity** tag.

Table 33-95 priceValidity

Field Name	Description
startValidityMode	Defines the start validity mode: • **FOREVER** • **IMMEDIATE** • **ABSOLUTE** • **RELATIVE** • **FIRST_USAGE** • **PERIOD_START**
endValidityMode	Defines the end validity mode: • **NEVER** • **ABSOLUTE** • **RELATIVE_TO_START** • **RELATIVE_TO_PERIOD_START**
validityRange	Defines the validity range in the following format: *YYYYMMDDTHHMMSS* or *YYYYMMDDTHHMMSS*.
relativeStartOffset	To make the amount valid in any number of accounting cycles, billing cycles, months, days, hours, minutes, or seconds from the event occurrence. For example, if you specify two accounting cycles, the amount will be valid after two accounting cycles from when the event occurs.
relativeEndOffset	The amount validity ends relative to the validity start time. Enter the number of accounting cycles, billing cycles, months, days, calendar days, hours, minutes, or seconds from the validity start time when the validity ends.

Bundle

This section describes the XML tags that you use to configure bundles with the XML pricing interface.

Bundles group together charge offers and discount offers belonging to the same service. See "Creating Bundles".

Copyright © 2017, 2026, Oracle and/or its affiliates.

bundledProductOffering

This is the main tag for configuring a bundle in PDC. Table 33-96 describes the fields in the **bundledProductOffering** tag.

Table 33-96 bundledProductOffering

Field Name	Description
name	Unique name of the bundle. It is VARCHAR2(255).
description	Description of the bundle. It is VARCHAR2(255).
internalId	Unique ID internally generated by PDC while committing the pricing configuration. It is unique for all bundles, charge offers, discount offers, and configurations in PDC, and cannot be the same for any other pricing or configuration entity.
pricingProfileName	The pricing profile determines which features are supported for the charge. PDC determines the **pricingProfileName** after the charge category and charge type are selected. Possible values are: • **Convergent Usage** • **Product Offering** • **Standard ChargeSharing** • **Standard Discount** • **Subscription ChargeSharing** • **Subscription Discount** • **Subscription**
priceListName	This is configured as **Default**.
obsolete	Boolean flag to obsolete the promoted (or active) objects.
timeRange	(Optional) Set the purchase start and end dates to specify when the bundle can be purchased in the following format: `[YYYYMMDDTHHMMSS/YYYYMMDDTHHMMSS]` The default is **0/inf**. For the start date, you can enter Immediately, or enter a specific start date and time. For the end date, you can enter **inf** (never) or a specific end date and time. The purchase period excludes the end date. For example, a bundle with a start date of May 1 and an end date of May 10 is available from May 1 through May 9.
productSpecName	The service type to which the charge offer applies.
assignedType	The type of configuration attribute specification: **PRODUCT** or **CUSTOMER**.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-96 (Cont.) bundledProductOffering

Field Name	Description
billOnPurchase	(Optional) Boolean flag for immediately generating a bill for any purchase fees when the customer purchases the bundle. This does not generate a bill for any other charges, such as recurring, usage, or cancellation fees.
customization	(Optional) Whether a CSR can change a customer's default price and effective time period of a charge or discount offer: • **PROHIBITED**: Prevents CSRs from changing the price and the effective period of a bundle. • **REQUIRED**: Forces CSRs to change the price and the effective period of a bundle. • **OPTIONAL**: Gives CSRs the option to change the price and the effective period of a bundle. This is the default.
firstUsageActivation	(Optional) Boolean flag for activating all charge offers within a bundle on first usage: • **true**: All charge offers in a bundle are activated when the first charge offer is activated. • **false**: When one charge offer in a bundle is activated on first usage, the other charge offers in the bundle are not activated. This is the default. See "Activating Offers in Bundles" for information and examples.
groupBalanceElements	(Optional) Boolean flag to synchronize the start date of all balance elements to the date of first usage when one of them is activated for the first time. This setting applies only to those balance elements that are configured to start on first usage, and only to those first-usage balance elements that are granted by the offers in the bundle.
bundledProductOfferingItem	Details about a charge offer or discount offer inside the bundle. For example it can include the **purchaseStart**, **purchaseEnd**, **usageStart**, **usageEnd**, **cycleStart**, and **cycleEnd** fields. See "bundledProductOfferingItem".

bundledProductOfferingItem

This is the main tag for configuring the item associated with the charge offer or discount offer in PDC. Table 33-97 describes the fields in the **bundledProductOfferingItem** tag.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-97 bundledProductOfferingItem

Field Name	Description
addOnOfferingValidityMode	(Optional) Adding this field specifies that the charge offer can be purchased as an add-on product after the bundle has been purchased. The validity mode for the add-on charge offer can be one of the following: • **BASE_OFFER**: Aligns the validity dates with the base charge offer. When set to **BASE_OFFER**, you must also include the **baseChargeOfferingName** field. • **BASE_OFFER_EXPIRY_FIRST**: Aligns the validity dates with the active base charge offer that expires first. • **BASE_OFFER_EXPIRY_LAST**: Aligns the validity dates with the active base charge offer that expires last. • **BASE_OR_ADD_ON_OFFER_EXPIRY_FIRST**: Aligns the validity dates with the active charge offer that expires first. • **BASE_OR_ADD_ON_OFFER_EXPIRY_LAST**: Aligns the validity dates with the active charge offer that expires last. See "Configuring Add-On Charge Offers In Bundles" for information and examples.
baseChargeOfferingName	The name of the base charge offer on which to set the validity dates for an add-on offer. **Note:** Include this field only when **addOnOfferingValidityMode** is set to **BASE_OFFER**.
chargeOfferingName	Displays the name of the charge offer or discount offer. This field is not editable.
cycleChargeAdjustment	(Optional) The adjustments to apply to the offer's recurring charges. The default is **0**.
gracePeriodOffset	(Optional) The length of the grace period during which an additional offer can be purchased as an extension of an existing subscription. The default is **0**. **Note:** This field applies only if **purchaseMode** is set to **EXTEND** or **OVERWRITE**. See "Purchasing the Same Offer in a Bundle Multiple Times" for information and examples.
gracePeriodUnit	(Optional) The unit of the grace period in **gracePeriodOffset**. The valid values are **SECOND**, **MINUTE**, **HOUR**, and **DAY**. The default is **DAY**. **Note:** This field applies only if **purchaseMode** is set to **EXTEND** or **OVERWRITE**. See "Purchasing the Same Offer in a Bundle Multiple Times" for information and examples.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-97 (Cont.) bundledProductOfferingItem

Field Name	Description
prorationBasedOn	The number of days in a month to use for calculating a prorated cycle charge or discount in an offer: • **PRORATE_30_DAYS**: Prorated charge or discount is based on a 30-day month. • **PRORATE_DAYS_IN_MONTH**: Prorated charge or discount is based on the actual number of days in the month, such as 30 for June or 31 for May. • Not Set or missing: Prorated according to the system-wide setting in the CM **pin.conf** file. See "Examples of 30-Day-Based Proration" in *BRM Configuring and Running Billing*. See "Setting Proration for Offers in a Bundle" for more information.
purchaseChargeAdjustment	(Optional) The adjustment to apply to the offer's purchase and cancel charges. The default is **0**.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-97 (Cont.) bundledProductOfferingItem

Field Name	Description
purchaseMode	(Optional) What happens if customers purchase the same charge offer or discount offer more than once. Possible values are: • **NEW_INSTANCE**: The additional offer is purchased as a new, unrelated subscription. This value is the default. • **OVERWRITE**: The additional offer replaces the existing subscription. • **LONGEST_DATE**: The additional offer is purchased as an extension to the existing subscription if it is purchased within the grace period. The new balance impacts are added to the existing balance group. The validity period for the balance is then set to either the old validity end date or the new validity end date, whichever is later. • **LONGEST_DATE_NEW_SUB_BAL**: The additional charge offer is purchased as an extension to the existing subscription if it is purchased within the grace period. The new balance impacts are added to a new balance group. The validity period for the new balance starts on the purchase date and the end is set to either the old validity end date or the new validity end date, whichever is later. • **EXTEND**: The additional offer is purchased as an extension to the existing subscription if it is purchased within the grace period. The new balance impacts are added to the existing balance group. The validity period for the balance is then set by adding the new validity period to whatever remains of the old validity period. • **EXTEND_NEW_SUB_BAL**: The additional charge offer is purchased as an extension to the existing subscription if it is purchased within the grace period. The new balance impacts are added to a new balance group. The validity period for the new balance starts on the end date for the old balance's validity and the new end date is set by adding the new validity period to whatever remains of the old validity period. **Note:** The following values can be applied to one-time charges only: **OVERWRITE**, **LONGEST_DATE**, **LONGEST_DATE_NEW_SUB_BAL**, **EXTEND**, and **EXTEND_NEW_SUB_BAL**. These values cannot be applied to cycle or usage charges. See "Purchasing the Same Offer in a Bundle Multiple Times" for information and examples.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-97 (Cont.) bundledProductOfferingItem

Field Name	Description
quantity	(Optional) The number of charge offers or discount offers included in the bundle. The default is **1**. For example, if a bundle for a cable service includes a charge offer that provides one set-top box and you want to include three set-top boxes with the cable service, enter 3 for the charge offer in that bundle.
renewalMode	(Optional) How to align recurring charge offers if a customer suspends and then reactivates a bundle: • **true**: Aligns the recurring charge offer as follows: – When **cycleFeeFlag** is set to **0**, the cycle aligns with the billing date. – When **cycleFeeFlag** is set to **1** and **purchaseCycleDOM** is not set, the cycle aligns with the reactivation date. – When **cycleFeeFlag** is set to **1** and **purchaseCycleDOM** is set, the cycle aligns with either **purchaseCycleDOM** or the reactivation date, whichever is later. • **false**: Aligns the recurring charge offer according to the setting in **purchaseCycleDOM**. This is the default. See "Setting Charge Offer Cycle Alignment for Reactivated Bundles" for information and examples.
status	(Optional) The initial status of the product offering: **1** (active) or **0** (inactive). The default is **1**.
statusCode	(Optional) A customer-defined reason code if a product offering's status is inactive.
usageChargeAdjustment	(Optional) The adjustments to apply to an offer's usage charge. The default is **0**.

Subscription Terms

This section describes the XML tags that you use to configure subscription terms with the XML pricing interface.

Subscription terms specify a contract's commitment period, such as 1 month, 1 year, or 2 years, and capture details about the term, such as the early termination penalty, auto contract renewal, and so on. For more information, see "Creating Subscription Terms".

terms

Table 33-98 describes the fields in the **terms** tag.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-98 terms

Field Name	Description
name	A unique name for the term to create.
description	A short description of the subscription term.
internalId	A unique ID internally generated by PDC while committing the pricing configuration. It cannot be the same as any other pricing or configuration entity.
pricingProfileName	The pricing profile specifies which features are supported for the term. Possible values are: • **Convergent Usage** • **Product Offering** • **Standard ChargeSharing** • **Standard Discount** • **Subscription ChargeSharing** • **Subscription Discount** • **Subscription**
priceListName	This is configured as **Default**.
obsolete	Boolean flag to obsolete the promoted (or active) objects.
termLength	The length of the commitment period. For example, for a 6-month commitment period, you would enter 6.
termUnit	The unit of the commitment period: **DAY**, **MONTH**, or **YEAR**. For example, for a 6-month commitment period, you would enter MONTH.
earlyTerminationOption	Whether to allow early cancellation of the contract and whether to apply an early termination fee. Possible values are: • **ALLOWED_NO_PENALTY**: Contracts can be canceled early, and customers will not be charged an early termination fee. • **ALLOWED_WITH_PENALTY**: Contracts can be canceled early, but customers will be charged an early termination fee for doing so. • **NOT_ALLOWED**: Contracts cannot be canceled early.
terminationGraceLength	The length of the grace period, during which the early termination fee is waived. For example, for a 10-day grace period, you would enter 10. **Note**: This value must be less than **termLength**.
terminationGraceUnit	The unit of the grace period: **DAY**, **MONTH**, or **YEAR**. For example, for a 10-day grace period, you would enter DAY.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-98 (Cont.) terms

Field Name	Description
minimumContractLength	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. **Note:** This value must be less than **termLength**, but greater than **terminationGraceLength**.
minimumContractUnit	The unit of the minimum contract length, such as **DAY**, **MONTH**, or **YEAR**.
penaltyAmount	The amount of the penalty.
penaltyBalanceElementNumCode	The balance element numeric code for the penalty amount, such as 840 for US Dollars.
penaltyRevenueGlid	The GLID to associate with the penalty amount.
penaltyOptions	The type of early termination fee 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.
autoRenew	Boolean flag specifying whether to automatically renew the contract: **true** or **false**. The default is **false**.

Time Model

This section describes the XML tags that you use to configure time models with the XML pricing interface.

A time model is a set of time periods that are used to set up different prices in a charge. Each time period consists of one or more time segments. A time segment can be represented using days of the week, days of the month, time of day, months of the year, or special days defined in a calendar. For more information, see "Charging Based on Date and Time".

timeModel

This is the main tag for configuring a time model in PDC. Table 33-99 describes the fields in the **timeModel** tag.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-99 timeModel

Field Name	Description
name	Unique name for the time model.
pricingProfileName	The pricing profile determines which features are supported for the time model. Possible values are: • **Convergent Usage** • **Product Offering** • **Standard ChargeSharing** • **Standard Discount** • **Subscription ChargeSharing** • **Subscription Discount** • **Subscription**
description	Description for the time model.
priceListName	This is configured as **Default**.
obsolete	Boolean flag to obsolete the promoted (or active) objects.
usesSpecialDayCalendar	It is an optional field. Specifies a special day calendar in this time model. If included, the time model must include at least one time period with a special day calendar. Otherwise, the time model validation will fail.
validityPeriod	Contains the time period details along with the time segment details.
validFrom	The start date.

timePeriod

Table 33-100 describes the fields in the **timePeriod** tag.

Table 33-100 timePeriod

Field Name	Description
tagName	Each time period within a time model should have a unique name.

timeSegment

Table 33-101 describes the fields in the **timeSegment** tag.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-101 timeSegment

Field Name	Description
timeSpecification	Displays the time segments for the time periods. A time segment defines the specific times in the time period. A time segment can include: • Days of the week • Time of day • Months of the year • Days of the month • Special days in a calendar
daysOfWeek	The days of the week for the time segment. To specify all days of the week, enter **Any Day of the Week**. To include specific days, enter one or more days. It has nested day tags.
timeOfDay	The time interval for the time segment. To set the time interval to 24 hours of the day, enter **All Day**. To set a specific time, enter a start and end time.
monthsOfYear	The months of the year for the time segment. To specify all months, enter **Any Month of the Year**. To include specific months, enter one or more months. It has nested month tags.
daysOfMonth	The days of the month for the time segment. To specify all days of the month, enter **Any Day of the Month**. To include specific days, enter one or more days.
holiday	Select to use special days defined in the special day calendar. You cannot have both normal days and special days included in the time period. They are mutually exclusive.

Charge Selector

This section describes the XML tags that you use to configure charge selectors with the XML pricing interface.

A charge selector is used to charge different fees for the same event based on the value of one or more attributes, such as subscriber (account), service, event, and custom rules (if applicable). For more information, see "About Selectors".

chargeSelector

Table 33-102 describes the fields in the **chargeSelector** tag.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-102 chargeSelector

Field Name	Description
name	Name of the charge selector. The name must be unique across all USC, APN, and charge selectors.
internalId	Unique ID internally generated by PDC while committing the pricing configuration. It cannot be the same as any other pricing or configuration entity.
description	Description for the charge selector.
eventName	The event type to which the charge selector applies.
pricingProfileName	The pricing profile for the charge selector. Only charges associated with the selected profile can be used as the result of each charge selector rule.
CandidateSelectionMode	(Optional) Controls selector evaluation. Supported values are **FIRST_MATCH**, **MIN_IMPACT**, and **MAX_IMPACT**. If the element is absent, ECE uses **FIRST_MATCH**. For selection behavior, see "How Impact-Based Candidate Selection Works" in *ECE Implementing Charging*.
CandidateSelectionBalanceElementCode	Identifies the currency or noncurrency balance element that ECE compares for **MIN_IMPACT** and **MAX_IMPACT**. Use **None** only with **FIRST_MATCH**.
priceListName	This is configured as **Default**.
obsolete	Boolean flag to obsolete the promoted (or active) objects.
stereoType	Type of object: **CHARGE_RATE_PLAN_SELECTOR**.
productSpecName	The service to which the charge selector applies.
eventSpecName	The event type to which the charge selector applies. Enter a fold event type if you are creating the charge selector for a fold charge.
customerSpecName	Tag used for "Account" as a service.
rumName	Name of the RUM.
measuredBy	The RUMs to which the charge selector applies. RUMs specify the units used to measure an event and how to calculate the measurement.

ruleAttributes

Use the **ruleAttributes** tag to select the subscriber, service, and event fields to use in the charge selector rules. Table 33-103 describes the fields in **ruleAttributes** tag.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-103 ruleAttributes

Field Name	Description
selectorOptions	Possible values: • **Zoning Not Used** • **Use Value Map** • **Use Zone Model**
modelData	Key-value pairs. For example: <modelData> <key>ZONING_OPTION</key> <value>ZONING_NONE</value> </modelData>

rules

Use the **rules** tag to define charge selector rules. To create rules for a charge selector, you must first specify the subscriber, service, or event attributes to use in the rules. Table 33-104 describes the fields in the **rules** tag.

Table 33-104 rules

Field Name	Description
validityPeriod	It indicates when the rules in the charge selector are valid. The default effective period starts immediately and never ends.
priority	The priority, starting with 1. The rules are evaluated in order of priority.
name	Unique name for the rule.
fieldName	An operator and value. The operator can be: • Equals or Not Equals • Matches or Not Matches • In or Not In
fieldToValueExpression	It has these fields: **operation**, **seperator**, **fieldName**, **fieldValue**, and **fieldKind**. For example: <fieldToValueExpression> <operation>EQUAL_TO</operation> <seperator>;</seperator> <fieldName>TelcoGsmTelephony.NAME</ fieldName> <fieldValue>abc</fieldValue> <fieldKind>PRODUCT_SPEC_FIELD</ fieldKind> </fieldToValueExpression>

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-104 (Cont.) rules

Field Name	Description
impactCategory	Enter the impact category to use for determining the charge.
chargeRatePlanName	The item to find, configure, or modify the charge used when the rule is true. • Search to add an existing charge • Create to configure a new charge • Edit to modify the current charge The charge must contain a price selector that associates the results impact category with a pricing instance.
balanceElementNumCode	Numeric code for the balance element.
glid	The required GL information for a given balance impact.
result	It has resultName, chargeSelector name, and resultData with key-value pairs.

Generic Selector

This section describes the XML tags that you use to configure generic selectors with the XML pricing interface.

A generic selector is a set of rules that map service and event attributes and expressions to a result. You use a generic selector in a charge to apply prices based on the rule results. For more information, see "About Selectors".

genericSelector

Table 33-105 describes the fields in the **genericSelector** tag.

Table 33-105 genericSelector

Field Name	Description
name	Name for the generic selector.
internalId	Unique ID internally generated by PDC while committing the pricing configuration. It cannot be the same as any other pricing or configuration entity.
description	Description for the generic selector.
productSpecName	(Optional) If you plan to use service attributes to create generic selector rules, enter the desired service.
eventSpecName	(Optional) If you plan to use event attributes to create generic selector rules, enter the desired event.
pricingProfileName	Pricing profile for the generic selector. Only charges associated with the selected profile can be used as the result of each generic selector rule.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-105 (Cont.) genericSelector

Field Name	Description
CandidateSelectionMode	(Optional) Specifies **FIRST_MATCH**, **MIN_IMPACT**, or **MAX_IMPACT**. If omitted, the Generic Selector uses the first matching result branch. For selection behavior, see "How Impact-Based Candidate Selection Works" in *ECE Implementing Charging*.
CandidateSelectionBalanceElementCode	Specifies the balance element used to compare matching result branches for **MIN_IMPACT** and **MAX_IMPACT**. Non-target balance element impacts do not participate in comparison.
priceListName	This is configured as **Default**.
obsolete	Boolean flag to obsolete the promoted (or active) objects.
stereoType	Type of object: **GENERIC_SELECTOR**.

ruleAttributes

Table 33-106 describes the fields in the **ruleAttributes** tag.

Table 33-106 ruleAttributes

Field Name	Description
field	Displays the fields that you have selected to define generic selector rules. The same fields are used for all the rules in the generic selector. You can override the default values and set specific values in the rule to limit the fields that you want used in a rule.
valueType	Select the type of value that the field contains. • **Single Value**: The field contains a single value. • **Value of Another Field**: The field contains the value of another field. If you select this option, in the Default column, select the other field.
operator	Displays the relationship between the rule's default field value and the value in the subscriber, service, or event field. The equal sign appears next to the field names when you define the rules. The value of the field in the subscriber, service, or event must match the value of the field in the rule.
default	Enter a default value for the field. An asterisk (*) specifies the field can contain any value. If the field is limited to predefined values, a list of values appears. Select a value from the list.

rules

Table 33-107 describes the fields in the **rules** tag.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-107 rules

Field Name	Description
validityPeriod	It indicates when the rules in the generic selector are valid. The default effective period starts immediately and never ends.
name	Name for the rule.
priority	The priority order, starting with 1. The rules are evaluated in order of priority.
fieldName	Enter an operator and enter a value. Operator can be: • Equals or Not Equals • Matches or Not Matches • In or Not In For example: <fieldToValueExpression> <operation>EQUAL_TO</operation> <seperator>;</seperator> <fieldName>EventDelayedSessionTelcoGsm.NUMBER_OF_UNITS</fieldName> <fieldValue>*</fieldValue> <fieldKind>EVENT_SPEC_FIELD</fieldKind> </fieldToValueExpression>
result	The result is used in a charge to guide to a price. Provide a string value to map to this rule. For example: • If the general selector rule determines that this is a Closed User Group (CUG) call, you could name the result ClosedUserGroup. • If the general selector rule determines that this is a call to friends and family, you could name the result Friends&Family. • If the general selector rule determines that this is a call with low QoS, you could name the result LowQoS.

Price Selector

This section describes the XML tags that you use to configure price selectors with the XML pricing interface.

A price selector associates event, service, and account attributes with prices. You use a price selector in a charge to apply prices based on price selector rules. For more information, see "About Selectors".

priceSelector

Table 33-108 describes the fields in the **priceSelector** tag.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-108 priceSelector

Field Name	Description
name	Unique name for the price selector.
description	Description for the price selector.
internalId	Unique ID internally generated by PDC while committing the pricing configuration. It cannot be the same as any other pricing or configuration entity.
productSpecName	(Optional) The service for which you want to create the price selector.
eventSpecName	(Optional) The event for which you want to create the price selector rules.
customerSpecName	Tag used for "Account" as a service.
pricingProfileName	The pricing profile for the generic selector. Only charges associated with the selected profile can be used as the result of each generic selector rule.
priceListName	This is configured as **Default**.
obsolete	Boolean flag to obsolete the promoted (or active) objects.
stereoType	Type of object: **PRICE_MODEL_SELECTOR**.

ruleAttributes

Table 33-109 describes the fields in the **ruleAttributes** tag.

Table 33-109 ruleAttributes

Field Name	Description
field	Displays the fields that you have selected to define price selector rules. The same fields are used for all rules in the price selector. You can override the default values and set specific values in the rule to limit the fields that you want used in a rule.
valueType	The type of value that the field contains: • **Single Value**: The field contains a single value. • **Value of Another Field**: The field contains the value of another field. If you enter this option, enter the name of the field in the **default** field.
operator	Displays the relationship between the rule's default field value and the value in the subscriber, service, or event field. The value of the field in the subscriber, service, or event must match the value of the field in the rule.
default	Enter a default value for the field. An asterisk (*) specifies the field can contain any value. If the field is limited to predefined values, a list of values appears.

Copyright © 2017, 2026, Oracle and/or its affiliates.

rules

Table 33-110 describes the fields in the **rules** tag.

Table 33-110 rules

Field Name	Description
validityPeriod	The rules in the price selector are valid. The default effective period starts immediately and never ends.
name	Provide a name for the rule.
priority	The priority order, starting with priority 1. The rules are evaluated in order of priority.
price	The existing pricing instance to associate with the rule.
fieldName	Select an operator and enter a value. Operator can be: • Equals or Not Equals • Matches or Not Matches • In or Not In For example: <fieldToValueExpression> <operation>REGEX</operation> <seperator>;</seperator> <fieldName>Account.CUST_A.ACCOUNT_NO </fieldName> <fieldValue>abc</fieldValue> <fieldKind>CUSTOMER_SPEC_FIELD</ fieldKind> </fieldToValueExpression>

Discount Selector

This section describes the XML tags that you use to configure discount selectors with the XML pricing interface.

A discount selector consists of rules that associate account, service, and event attributes and custom rules with discounts. You use a discount selector to determine which discount to apply to an event. For more information, see "About Selectors".

discountSelector

Table 33-111 describes the fields in the **discountSelector** tag.

Table 33-111 discountSelector

Field Name	Description
name	Unique name for the discount selector.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-111 (Cont.) discountSelector

Field Name	Description
internalId	Unique ID internally generated by PDC while committing the pricing configuration. It cannot be the same as any other pricing or configuration entity.
description	Description for the discount selector.
productSpecName	(Optional) Service for which you want to create the discount selector.
eventSpecName	(Optional) Event for which you want to create the discount selector.
pricingProfileName	Pricing profile to associate with the discount selector.
priceListName	This is configured as **Default**.
obsolete	Boolean flag to obsolete the promoted (or active) objects.
stereoType	Type of object: **DISCOUNT_SELECTOR**.

ruleAttributes

Table 33-112 describes the fields in the **ruleAttributes** tag.

Table 33-112 ruleAttributes

Field Name	Description
field	Displays the fields in your discount selector rules. The same fields are used for all rules in the discount selector. You can override the default values and set specific values in the rule to limit the fields that you want used in a rule.
valueType	Type of value that the field contains: • **Single Value**: The field contains a single value. • **Value of Another Field**: The field contains the value of another field. If you enter this option, enter a field name in the **default** field.
operator	Displays the relationship between the rule's default field value and the value in the subscriber, service, event, custom rule, or profile field: **Equals**, **Matches**, or **Maps**.
default	Enter a default value for the field. An asterisk (*) specifies that the field can contain any value. If the operator is Equals or Maps to, an asterisk (*) specifies the field can contain any value. If the operator is Matches, a dot followed by an asterisk (.*) specifies the field can contain any value. If the field is limited to predefined values, a list of values appear. Select a value from the list. If Value of Another Field is selected for Value Type, the Select Field dialog box appears. In the Select Field dialog box, select the field you want.

Copyright © 2017, 2026, Oracle and/or its affiliates.

rules

Table 33-113 describes the fields in the **rules** tag.

Table 33-113 rules

Field Name	Description
validityPeriod	When the rules in the discount selector are valid. The default effective period starts immediately and never ends.
name	The name for the rule.
priority	The priority order of the rule, starting with priority 1.
discount	An existing discount to associate with the rule.
fieldToValueExpression	An operator and a value. The operator can be: • Equals or Not Equals • Matches or Not Matches • In or Not In For example: <fieldToValueExpression> <operation>REGEX</operation> <seperator>;</seperator> <fieldName>Account.CUST_A.ACCOUNT_NO</fieldName> <fieldValue>abc</fieldValue> <fieldKind>CUSTOMER_SPEC_FIELD</fieldKind> </fieldToValueExpression>

USC Selector

This section describes the XML tags that you use to configure USC selectors in the XML pricing interface.

A Usage Scenario (USC) selector contains rules that map existing impact categories to new impact categories based on usage attributes and custom rules. For more information, see "About Selectors".

USCSelector

Table 33-114 describes the fields in the **USCSelector** tag.

Table 33-114 USCSelector

Field Name	Description
name	Unique name for the USC selector. The name must be unique across USC, APN, and charge selectors.
description	Description for the USC selector.
internalId	Unique ID internally generated by PDC while committing the pricing configuration. It cannot be the same as any other pricing or configuration entity.
pricingProfileName	Pricing profile to associate with the USC selector.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-114 (Cont.) USCSelector

Field Name	Description
priceListName	This is configured as **Default**.
obsolete	Boolean flag to obsolete the promoted (or active) objects.
stereoType	Type of object: **USC_MAP**.

rulesAndConditions

This section is used to create rules to map existing impact categories to new impact categories. Table 33-115 describes the fields in the **rulesAndConditions** tag.

Table 33-115 rulesAndConditions

Field Name	Description
validityPeriod	(Optional) The start and end dates that specify a time period during which the rules in the USC selector are valid. The default effective period starts immediately and never ends. For example, if starts from Immediately, validFrom value is 0.
priority	The priority in numerical order, starting with 1. The rules are evaluated in order of priority. The priority numbers of all rules affected by the move are changed.
name	Name for the USC selector rule.
ruleData	Zone model to use for this USC selector rule: • **key**: ZONE_MODEL • **value**: Corresponding value of the zone model.
result	The new impact category to use for determining the price for the event. The inner tag **resultName** has the impact category name.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-115 (Cont.) rulesAndConditions

Field Name	Description
zoneImpactCategory	The operator and the existing zone impact categories that you want to map to the new impact category in this USC selector rule. The default value **.*** specifies that the rule applies for all zone impact categories. **Note**: You can use Not Matches only if you are using ECE for usage charging. For example: `<fieldToValueExpression>` `<operation>REGEX</operation>` `<seperator>;</seperator>` `<fieldName>AnyEvent.RETAIL_ZONE</fieldName>` `<fieldValue>.*</fieldValue>` `<fieldKind>EVENT_SPEC_FIELD</fieldKind>` `</fieldToValueExpression>`
usageClass	The operator and the existing usage class values to use for evaluating the rule. The default value **.*** specifies that the rule applies for all usage classes. **Note**: You can use Not Matches only if you are using ECE for usage charging. For example: `<fieldToValueExpression>` `<operation>REGEX</operation>` `<seperator>;</seperator>` `<fieldName>AnyEvent.USAGE_CLASS</fieldName>` `<fieldValue>CONF</fieldValue>` `<fieldKind>EVENT_SPEC_FIELD</fieldKind>` `</fieldToValueExpression>`
usageType	The operator and the existing usage type values to use for evaluating the rule. The default value **.*** specifies that the rule applies for all usage types. **Note**: You can use Not Matches only if you are using ECE for usage charging.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-115 (Cont.) rulesAndConditions

Field Name	Description
customRule	The operator and the existing custom rule to use for evaluating the rule. The default value **.*** specifies that the rule applies for all custom rules. **Note**: You can use Not Matches only if you are using ECE for usage charging. Example: `<crSelectorExpression>` `<rightOperand>CUST_RULE_CUSTOMER1</rightOperand>` `<operator>NONE</operator>` `</crSelectorExpression>`
serviceClass	The operator and the existing service class values to use for evaluating the rule. The default value **.*** specifies that the rule applies for all service classes. **Note**: You can use Not Matches only if you are using ECE for usage charging. For example: `<fieldToValueExpression>` `<operation>REGEX</operation>` `<seperator>;</seperator>` `<fieldName>AnyEvent.SERVICE_CLASS</fieldName>` `<fieldValue>DEF</fieldValue>` `<fieldKind>EVENT_SPEC_FIELD</fieldKind>` `</fieldToValueExpression>`

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-115 (Cont.) rulesAndConditions

Field Name	Description
serviceCode	The operator and the existing service code values to use for evaluating the rule. The default value **.*** specifies that the rule applies for all service classes. **Note**: You can use Not Matches only if you are using ECE usage charging. For example: `<fieldToValueExpression>` `<operation>REGEX</operation>` `<seperator>;</seperator>` `<fieldName>AnyEvent.SERVICE_CODE</fieldName>` `<fieldValue>FAX</fieldValue>` `<fieldKind>EVENT_SPEC_FIELD</fieldKind>` `</fieldToValueExpression>`
quantityValue	(Optional) Enter the maximum quantity of usage for the event, such as duration. The quantity of usage for the event must be less than or equal to the quantity value.
newUsageType	(Optional) The new usage type to assign to the event.

APN Selector

This section describes the XML tags that you use to configure Access Point Name (APN) selectors with the XML pricing interface.

An APN selector consists of rules that map APNs to new impact categories based on event attribute values. You use APN selectors to redefine the impact category for events that are associated with specific APNs. For more information, see "About Selectors".

APN Selectors

Table 33-116 describes the fields in the **apnSelectors** tag.

Table 33-116 APN Selectors

Field Name	Description
name	Unique name for the APN selector. The name must be unique across USC, APN, and charge selectors.
description	Description for the APN selector.
internalId	Unique ID internally generated by PDC while committing the pricing configuration. It cannot be the same as any other pricing or configuration entity.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-116 (Cont.) APN Selectors

Field Name	Description
pricingProfileName	Pricing profile to associate with the APN selector.
priceListName	This is configured as **Default**.
obsolete	Boolean flag to obsolete the promoted (or active) objects.
stereoType	Type of object: **APN_MAP**.

rules

Use the **rules** tag to create rules to associate specific APNs with new impact categories. Table 33-117 describes the fields in the **rules** tag.

Table 33-117 rules

Field Name	Description
validityPeriod	(Optional) The start and end dates that specify a time period during which the rules in the USC selector are valid. The default effective period starts immediately and never ends.
priority	The priority in numerical order, starting with 1. The rules are evaluated in order of priority.
name	Name for the APN selector rule.
modelData	Zone model to use for this USC selector rule. • **key**: ZONE_MODEL • **value**: Corresponding value of the zone model.
APNName	The operator and the existing APN name to use for evaluating the rule. The default value **.*** specifies that the rule applies for all APNs. **Note**: You can use Not Matches only if you are using ECE for usage charging. For example: `<fieldToValueExpression>` `<operation>REGEX</operation>` `<seperator>;</seperator>` `<fieldName>AnyEvent.ACCESS_POINT_NAME</fieldName>` `<fieldValue>.*</fieldValue>` `<fieldKind>EVENT_SPEC_FIELD</fieldKind>` `</fieldToValueExpression>`

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-117 (Cont.) rules

Field Name	Description
serviceCode	The operator and the existing service codes to use for evaluating the rule. The default value **.*** specifies that the rule applies for all service codes. **Note**: You can use Not Matches only if you are using ECE for usage charging. For example: `<fieldToValueExpression>` `<operation>REGEX</operation>` `<seperator>;</seperator>` `<fieldName>AnyEvent.SERVICE_CODE</fieldName>` `<fieldValue>FAX</fieldValue>` `<fieldKind>EVENT_SPEC_FIELD</fieldKind>` `</fieldToValueExpression>`
zoneImpactCategory	The operator and existing zone impact categories associated with the APNs. The default value **.*** specifies that the rule applies for all zone impact categories. For example: `<fieldToValueExpression>` `<operation>REGEX</operation>` `<seperator>;</seperator>` `<fieldName>AnyEvent.RETAIL_ZONE</fieldName>` `<fieldValue>.*</fieldValue>` `<fieldKind>EVENT_SPEC_FIELD</fieldKind>` `</fieldToValueExpression>`
result	The new impact category to associate with the APNs. The inner tag **resultName** has the impact category name.

Package

This section describes the XML tags that you use to configure packages with the XML pricing interface.

A package consists of one or more bundles that you make available to your customers for purchase. For more information, see "Creating Packages".

Copyright © 2017, 2026, Oracle and/or its affiliates.

Package

Table 33-118 describes the fields in the **package** tag.

Table 33-118 Package

Field Name	Description
name	Unique name given for a package.
description	Description for the package.
internalId	Unique ID internally generated by PDC while committing the pricing configuration. It cannot be the same as any other pricing or configuration entity.
pricingProfileName	Pricing profile to associate with the package.
priceListName	This is configured as **Default** by default.
obsolete	Boolean flag to obsolete the promoted (or active) objects.
billOnPurchase	(Optional) Boolean flag for billing customers immediately for a purchase, even if the customer's billing cycle has not ended. When a customer purchases a package that is flagged for on-purchase billing, a bill is generated immediately for the purchase fees associated with the package. This does not generate a bill for any other charges, such as recurring, usage, or cancellation fees.
firstUsageActivation	(Optional) Boolean flag for activating all charge offers within a package on first usage: • **true**: All charge offers in a package are activated on first usage. • **false**: When one charge offer in a package is activated on first usage, the other charge offers in the package are not activated. This is the default. See "Activating Offers in Packages" for information and examples.
customerSpecPackageItem	Tag used for holding the bundle association attributes under "Account" in a package.
productSpecPackageItem	Tag used for holding the bundle association attributes under "Service*" (such as TelcoGsmSms, TelcoGsmSms, and so on) in a package.
balanceSpecification	By default, balances for all services are tracked in the account balance group. To track balances for specific services in the package, you associate the services with their own balance groups. For each balance group, you can specify credit limits, credit thresholds, and consumption rules for balance elements used by the services in that group. See "Setting Loan Thresholds for Packages" for examples.
subscriptionTermsIID	Internal ID for the subscription terms object.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-118 (Cont.) Package

Field Name	Description
subscriptionTermsName	Name for the subscription terms object.
productSpecCharacteristic	Key-value pair for defining special characteristics in a product.

Package List

This section describes the XML tags that you use to configure package lists using the XML pricing interface.

A package list enables you to group related packages that you can sell to your customers. For more information, see "Creating Package Lists".

Package List

Table 33-119 describes the fields in the **packageList** tag.

Table 33-119 Package List

Field Name	Description
name	Name for the package list.
description	Description for the package list.
internalId	Unique ID internally generated by PDC while committing the pricing configuration. It cannot be the same as any other pricing or configuration entity.
pricingProfileName	Pricing profile to associate with the package list.
priceListName	This is configured as **Default** by default.
obsolete	Boolean flag to obsolete the promoted (or active) objects.
type	Type of packages you want to add to your package list. The combination of the package list segment and type are case-sensitive and uniquely identify a package list when the package list is used in BRM.
segment	The customer segment that the package list is offered to. The combination of the package list segment and type are case-sensitive and uniquely identify a package list.
packageListStatus	Whether you want the package list to be available for purchase as soon as it is added to the database: • **Active**: Indicates that customers can purchase packages in the list as soon as the list is added to your system. • **Inactive**: Indicates that the list is not visible to customers, and customers cannot purchase packages from the list.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Tax Selector

This section describes the XML tags that you use to configure tax selectors with the XML pricing interface.

A tax selector allows you to use attributes in **/account**, **/service**, **/event**, or **/profile** objects to set a tax rate. See "About Selectors" for more information.

taxSelector

Table 33-120 describes the fields in the **taxSelector** tag.

Table 33-120 taxSelector

Field Name	Description
name	Unique name of the tax selector.
description	Description of the tax selector.
pricingProfileName	Pricing profile to associate with the tax selector.
priceListName	This is configured as **Default** by default.
obsolete	Boolean flag to obsolete the promoted (or active) objects.
stereoType	Set to **TAX_SELECTOR**.
productSpecName	(Optional) The service for which you want to create the tax selector.
eventSpecName	(Optional) The event for which you want to create the tax selector.
customerSpecName	(Optional) Tag used for "Account" as a service.
validityPeriod	Indicates when the rules in the tax selector are valid. The default effective period starts immediately and never ends.

validityPeriod

Table 33-121 describes the fields in the **validityPeriod** tag.

Table 33-121 validityPeriod

Field Name	Description
validFrom	This contains the start date of the validity period, or **0** to indicate that it has no start date.
rule	This contains the characteristics of the rule to be checked.

rule

Table 33-122 describes the fields in the **rule** tag.

Table 33-122 rule

Field Name	Description
name	This contains the name of the rule.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-122 (Cont.) rule

Field Name	Description
result	This contains one element, **resultName**, which contains the name of a tax code defined in the system.
fieldToValueExpression	Enter the information for the comparison, as shown below: `<fieldToValueExpression>` `  <operation>REGEX</operation>` `  <seperator>;</seperator>` `<fieldName>EventDelayedSessionTelcoGsm.CALLED_ID</fieldName>` `  <fieldValue>1800.*</fieldValue>` `  <fieldKind>EVENT_SPEC_FIELD</fieldKind>` `</fieldToValueExpression>`

Tax Exemption Selector

This section describes the XML tags that you use to configure tax exemption selectors with the XML pricing interface.

A tax exemption selector allows you to use attributes in **/account**, **/service**, **/event**, or **/profile** objects to set a tax exemption rate. See "About Selectors" for more information.

taxExemptionSelector

Table 33-123 describes the fields in the **taxExemptionSelector** tag.

Table 33-123 taxExemptionSelector

Field Name	Description
name	Unique name of the tax exemption selector.
description	Description of the tax exemption selector.
pricingProfileName	Pricing profile to associate with the tax exemption selector.
priceListName	This is configured as **Default** by default.
obsolete	Boolean flag to obsolete the promoted (or active) objects.
stereoType	Set to **TAX_EXEMPTION_SELECTOR**.
productSpecName	The service for which you want to create the tax exemption selector.
eventSpecName	The event for which you want to create the tax exemption selector.
customerSpecName	Tag used for "Account" as a service.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-123 (Cont.) taxExemptionSelector

Field Name	Description
validityPeriod	Indicates when the rules in the tax exemption selector are valid. The default effective period starts immediately and never ends.

validityPeriod

Table 33-124 describes the fields in the **validityPeriod** tag.

Table 33-124 validityPeriod

Field Name	Description
validFrom	This contains the start date of the validity period, or **0** to indicate that it has no start date.
rule	This contains the characteristics of the rule to be checked.

rule

Table 33-125 describes the fields in the **rule** tag.

Table 33-125 rule

Field Name	Description
name	This contains the name of the rule.
result	This contains one element, **resultName**, which contains the name of a tax exemption code defined in the system.
fieldToValueExpression	Enter the information for the comparison, as shown below: `<fieldToValueExpression>` `  <operation>EQUAL_TO</operation>` `  <seperator>;</seperator>` `<fieldName>TelcoGprs.GPRS_INFO.ROUTIN G_AREA</fieldName>` `  <fieldValue>National</fieldValue>` `  <fieldKind>PRODUCT_SPEC_FIELD</ fieldKind>` `</fieldToValueExpression>`

Price Tag Setup Component

This section describes the XML tags that you use to configure price tags as setup components in the XML pricing interface.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Price tags let you apply special prices to charge offers or discount offers according to specified rules. You configure possible values and validity rules for price tags in **priceTagConfigs** setup components. After importing the setup components, you can associate price tags with charges and discounts and set their values. See

priceTagConfigs

This is the main tag for configuring a charge offer in PDC. Table 33-126 describes the fields in the **priceTagConfigs** tag.

Table 33-126 priceTagConfigs

Field Name	Description
name	Unique name of the price tag. The data type is VARCHAR2 (255).
description	Description of the price tag. This field is optional. The data type is VARCHAR2 (255).
tags	The container tag for a price tag within the configuration object.

tags

The **tags** tag represents the tag rules for the price tag. Table 33-127 describes the fields in the **tags** tag.

Table 33-127 tags

Field Name	Description
ruleType	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.
ruleValue	The valid values, in the format that corresponds to the rule: • If **<ruleType>** is set to **ANY**, the value is set to an empty string, which can be defined when adding a price tag to a charge offer or discount offer. • If **<ruleType>** 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 **<ruleType>** 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.
resourceId	The resource ID for the balance elements you can use the price tag with. For example, **Free Minutes**, **840** for US Dollars, or **0** to add the price tag to multiple balance elements. **Note:** You cannot use price tags with temporary balance elements. If you try to import a price tag with the resource ID of a temporary balance element, the **ImportExportPricing** utility will return an error.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 33-127 (Cont.) tags

Field Name	Description
resourceUnitType	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.
permittedNames	The container tag for the services that you can use the price tag with. Use ***** to use it with any service.

permittedNames

The **permittedNames** tag specifies the services that you can use a price tag with. Table 33-128 describes the fields in the **chargeEventMap** tag.

Table 33-128 permittedNames

Field Name	Description
name	The name of the service. For example, **TelcoGsmSms** or **Email**. The default is *****, which lets you use the price tag with all service types.
applicableToAllChildServices	Boolean flag to enable the price tag to be used with all child services of the service specified in **name**.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Part VI

Customization and Administration

This part describes how to customize and maintain Pricing Design Center (PDC) and Oracle Communications Billing and Revenue Management (BRM) product offerings.

This part contains the following chapters:

- Enabling Charging for Custom Events
- Using the PDC Web Service
- Working with Profiles
- Configuring Product Specification Attributes for Pricing Components
- Replicating Data Between PDC Systems

Copyright © 2017, 2026, Oracle and/or its affiliates.

34

Enabling Charging for Custom Events

Learn how to enrich custom events in Pricing Design Center (PDC) with information required by Oracle Communications Elastic Charging Engine (ECE) to process usage charging requests from the network.

Topics in this document:

- Overview of Enriching Event Definitions
- About Charging Operation Types
- About Charging Operation Type Versions
- About Mapping Network Attributes to Event Attributes
- Before Enriching Event Definitions
- Enriching Event Definitions
- Editing Event Definitions

Overview of Enriching Event Definitions

Oracle Communications Billing and Revenue Management (BRM) includes a set of commonly used event definitions, such as the events used by GSM and GPRS services. If you create custom events and services by using BRM Developer Center, you need to use PDC to enrich the event definitions with additional information required by ECE.

You do the following to enrich an event definition:

- Add **USER_IDENTITY** and **CALLED_ID** attributes. The **USER_IDENTITY** attribute is required for all events. The **CALLED_ID** attribute is required only if you use zoning or other charging scenarios that are based on the call destination.
- Add transient attributes if required. These are attributes that are used specifically for usage charging and do not need to be stored persistently in the event for rerating. For example, you can add the CELL_ID attribute to retrieve cell load information, but the CELL_ID value does not need to be stored in the event object.
- Add usage charging operation types (for example, initiating a session-based charge for an event). See "About Charging Operation Types" for more information.
- Define the network mapping for the events; for example, if you are using Diameter Gateway, you map the request attributes that are received from the Diameter network to event attributes. This mapping is used by the ECE charging client at runtime to populate the event. See "About Mapping Network Attributes to Event Attributes" for more information.

About Charging Operation Types

A charging operation type is a type of operation that ECE uses for usage charging; for example, the **Initiate** operation type is used for initiating a session-based charge, and the **Price_Enquiry** operation type is used for inquiring about price information. Each charging operation type requires a set of event attributes to be specified.

Copyright © 2017, 2026, Oracle and/or its affiliates.

When you add the charging operation type in the event definition, you associate the charging operation type with the group of event attributes that are necessary to perform the charging operation on the event. For example:

- The attributes for session-based charging are associated with the **Initiate**, **Update**, and **Terminate** operation types.
- The attributes for event-based charging are associated with the **Debit**, **Refund**, and **Price_Enquiry** charging operation types.

Table 34-1 lists the charging operation types that are used in ECE.

Table 34-1 Charging Operation Types Used in ECE

Operation Type	Description
Initiate	Initiates a session-based charging operation.
Update	Modifies a session-based charging operation.
Terminate	Terminates a charging operation.
Cancel	Cancels a session-based charging operation.
Refund_Amount	Refunds a specific amount to a specific balance.
Refund_Unit	Refunds a calculated amount, based on units consumed, to the impacted balances.
Debit_Amount	Debits a specific amount to a specific balance.
Debit_Unit	Debits a calculated amount, based on units consumed, to the impacted balances.
Price_Enquiry	Generates a price estimation without any balance reservations occurring. It is used when there is a low probability of receiving a charging request. For example, Price_Enquiry might be called to get the price of an event charge to display in a content portal.
Start_Accounting	Starts tracking usage without incurring balance impacts.
Update_Accounting	Continues tracking usage without incurring balance impacts.

See "Enriching Event Definitions" for information on how to add charging operation types and associate them with the event attributes.

About Charging Operation Type Versions

A set of charging operation types has a version, which is indicated by the **<externalVersion>** element.

When you modify the charging operation types in PDC, use the same version number to replace the existing set of charging operation types with the updated ones. The updated charging operation types are published to ECE.

You increase the version number if you want to use both the existing and updated set of charging operation types. For example, when incrementally upgrading the network mediation software program (client), you may want both sets of charging operation types active at the same time. Both old and new versions of the charging operation types are stored in PDC and published to ECE.

You must update the version number in your mediation specification file to associate the usage request builder with the event definition to which it applies.

Copyright © 2017, 2026, Oracle and/or its affiliates.

The following is the syntax for the charging operation type version:

```
<opVersion>
   <externalVersion>version_number</externalVersion>
   <opTypes>
      ...
   </opTypes>
   <opTypes>
      ...
   </opTypes>
</opVersion>
```

where *version_number* is the version number for the set of charging operation types.

About Mapping Network Attributes to Event Attributes

The data that ECE needs for charging is sent to ECE as network attributes from clients. For example, the call destination might be sent to ECE as the **Service-Information.PS-Information.Called-Station-Id** attribute. To charge for usage, ECE needs to:

- Map network attributes to event attributes.
- Indicate whether the network attribute is from a 4G network or a 5G network:
  - For 4G networks, add the network attribute to the **<networkAttributeItem>** element. ECE rating uses this to calculate the value of the event attribute (specified by **<name>**), which will be stored in the field specified by **<persistedName>**.
  - For 5G networks, add the network attribute to the **<networkAttributeItem5g>** element. ECE rating uses this to calculate the value of the event attribute (specified by **<name>**), which will be stored in the field specified by **<persistedName5g>**.

> **Note**
>
> The **<persistedName5g>** element is used only with the HTTP Gateway. See "Connecting ECE to a 5G Client" in *ECE Implementing Charging*.

For example, the following shows how to map the **Service-Information.PS-Information.Called-Station-Id** attribute from a 5G network to the **CALLED_ID** event attribute. The calculated value of the **CALLED_ID** event attribute will be stored in the **my_called_id_c** field in the BRM database.

```
<simpleAttributeItem>
       <name>CALLED_ID</name>
       <description>CALLED_ID</description>
       <type>STRING</type>
       <persistedName5g>my_called_id_c</persistedName5g>
       <attributeItemSupportedBy>
            <targetApplicationSpecName>Pricing</targetApplicationSpecName>
            <targetApplicationSpecName>Convergent Charging</targetApplicationSpecName>
       </attributeItemSupportedBy>
            <networkAttributeItem5g>Service-Information.PS-Information.Called-Station-
Id</networkAttributeItem5g>
</simpleAttributeItem>
```

Each event attribute must specify where to store the definition. This can be in either in BRM or ECE cache. You specify the location using **<targetApplicationSpecName>** in the **<attributeItemSupportedBy>** element.

Copyright © 2017, 2026, Oracle and/or its affiliates.

If the attribute is defined in BRM, add **<targetApplicationSpecName>Billing</targetApplictionSpecName>**.

If the attribute is not defined in BRM and is created for ECE usage rating, use:

```
<attributeItemSupportedBy>
    <targetApplicationSpecName>Pricing</targetApplicationSpecName>
    <targetApplicationSpecName>Convergent Charging</targetApplicationSpecName>
</attributeItemSupportedBy>
```

If the definition is defined in BRM and can be used for real-time usage rating, add:

```
 <targetApplicationSpecName>Realtime Charging</targetApplicationSpecName>
```

If the definition is defined in BRM and can be used for batch rating, add:

```
<targetApplicationSpecName>Batch Charging</targetApplicationSpecName>
```

If the definition is defined in BRM and can be used for ECE usage rating, add:

```
<targetApplicationSpecName>Convergent Charging</targetApplicationSpecName>
```

Before Enriching Event Definitions

Before enriching an events definition:

- Create a new event subclass (for example, **/event/cloud/library**) in Developer Center. See *BRM Developer's Guide* for more information.
- When you create the event subclass, add your custom fields at the top level or under a substruct, such as USAGE_INFO. Ensure that the fields for storing the user identity and called ID (if you are using zoning in your charge offer) are added to the substruct in the event class.

  See the discussion about substructure in *BRM Developer's Guide* for more information.
- The service and event definitions are synchronized with PDC by running the **SyncPDC** utility. See "Creating Pricing Setup Components" for more information.
- The event is mapped to the corresponding service (for example, **/service/cloud/library**). See "Setting Up the Service-Event Map" for more information.

Enriching Event Definitions

To enrich an event definition:

1. Export the event definition from PDC into an XML file by running the following command:

   ImportExportPricing -export *FileNamePrefix* **-metadata EVENT_ATTRIBUTE_SPEC -n** "*eventname*"

   where *eventname* is the name of the event to be exported into an XML file.

   For example, this command exports the definition for EventCloudLibrary to the **EventCloudLibrary_export_metadata.xml** file:

Copyright © 2017, 2026, Oracle and/or its affiliates.

```
ImportExportPricing -export EventCloudLibrary -metadata EVENT_ATTRIBUTE_SPEC -n
EventCloudLibrary
```

2. Open the event definition file in a text editor.

   The following example shows the event definition for the EventCloudLibrary event:

```
<eventAttributeSpec>
   <name>EventCloudLibrary</name>
   <description>EventCloudLibrary</description>
   <internalId>c06de0cc-6105-4cd3-92c9-ad8011e5fe7b</internalId>

   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <attributeSpecSupportedBy>
      <targetApplicationSpecName>Billing</targetApplicationSpecName>
      <targetApplicationSpecName>Pricing</targetApplicationSpecName>
      <targetApplicationSpecName>Convergent Charging</targetApplicationSpecName>
      <targetApplicationSpecName>Realtime Charging</targetApplicationSpecName>
   </attributeSpecSupportedBy>
   <status>ENABLED</status>
   <className>/event/cloud/library</className>
   <complexAttributeItem>
      <name>USAGE_INFO</name>
      <description>USAGE_INFO</description>
      <type>STRUCT</type>
      <persistedName5g>cloudlibrary_usage_info_t</persistedName5g>
      <simpleAttributeItem>
         <name>MY_IDENTITY</name>
         <description>MY_IDENTITY</description>
         <type>STRING</type>
         <persistedName5g>my_identity_c</persistedName5g>
         <attributeItemSupportedBy>
            <targetApplicationSpecName>Pricing</targetApplicationSpecName>
            <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
         </attributeItemSupportedBy>
      </simpleAttributeItem>
      <simpleAttributeItem>
         <name>MY_CALLED_ID</name>
         <description>MY_CALLED_ID</description>
         <type>STRING</type>
         <persistedName5g>my_called_id_c</persistedName5g>
         <attributeItemSupportedBy>
           <targetApplicationSpecName>Pricing</targetApplicationSpecName>
           <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
         </attributeItemSupportedBy>
      </simpleAttributeItem>
      <simpleAttributeItem>
         <name>SERVICE_PROVIDER</name>
         <description>SERVICE_PROVIDER_C</description>
         <type>STRING</type>
         <persistedName5g>service_provider_c</persistedName5g>
         <attributeItemSupportedBy>
            <targetApplicationSpecName>Pricing</targetApplicationSpecName>
            <targetApplicationSpecName>Convergent Charging</
targetApplicationSpecName>
         </attributeItemSupportedBy>
         <length>60</length>
      </simpleAttributeItem>
      <simpleAttributeItem>
         <name>SERVICE_STATUS</name>
         <description>SERVICE_STATUS_C</description>
         <type>STRING</type>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
            <persistedName5g>service_status_c</persistedName5g>
            <attributeItemSupportedBy>
               <targetApplicationSpecName>Pricing</targetApplicationSpecName>
               <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
            </attributeItemSupportedBy>
            <length>60</length>
         </simpleAttributeItem>
         <simpleAttributeItem>
            <name>SERVICE_TYPE</name>
            <description>SERVICE_TYPE_C</description>
            <type>STRING</type>
            <persistedName5g>service_type_c</persistedName5g>
            <attributeItemSupportedBy>
               <targetApplicationSpecName>Pricing</targetApplicationSpecName>
               <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
            </attributeItemSupportedBy>
            <length>60</length>
         </simpleAttributeItem>
      </complexAttributeItem>
      <eventType>SESSION_USAGE</eventType>
      <baseEventSpec>EventCloud</baseEventSpec>
      <virtualColumn>false</virtualColumn>
      <unit>MONTHS</unit>
      <frequency>0</frequency>
   </eventAttributeSpec>
```

3. Add USER_IDENTITY and CALLED_ID (if you are using zoning in your product offerings) attributes as simple attribute items immediately under the **<className>** element by doing the following:

   a. Search for the following element:

      <className>*EventClassName***</className>**

      where *EventClassName* is the class name of the event.

      For example:

      ```
      <className>/event/cloud/library</className>
      ```

   b. Add the **USER_IDENTITY** and **CALLED_ID** attributes as simple attribute items immediately under the **<className>** element:

      ```
      <className>/event/cloud/EventUsage</className>
      <simpleAttributeItem>
         <name>USER_IDENTITY</name>
         <description>USER_IDENTITY</description>
         <type>STRING</type>
         <persistedName5g>PersistedNameofAttribute</persistedName5g>
         <attributeItemSupportedBy>
            <targetApplicationSpecName>Pricing</targetApplicationSpecName>
            <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
         </attributeItemSupportedBy>
      </simpleAttributeItem>

      <simpleAttributeItem>
         <name>CALLED_ID</name>
         <description>CALLED_ID</description>
         <type>STRING</type>
         <persistedName5g>PersistedNameofAttribute</persistedName5g>
         <attributeItemSupportedBy>
              <targetApplicationSpecName>Pricing</targetApplicationSpecName>
              <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
      ```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
   </attributeItemSupportedBy>
</simpleAttributeItem>
```

where *PersistedNameofAttribute* specifies the name of the BRM database table and column associated with the attributes used for storing the user identity and called ID; for example, cloudlibrary_usage_info_t.my_identity_c and cloudlibrary_usage_info_t.my_called_Id_c.

4. Add the event attributes that you use in ratable usage metric (RUM) expressions for **Initiate** or **Update** charging operations as top-level attributes under the **<className>** element.

   The following example shows the event attributes used in the RUM expressions for charging the EventCloudLibrary event:

```
<simpleAttributeItem>
   <name>DURATION</name>
   <description>DURATION</description>
   <type>UNIT_VALUE</type>
   <unitType>TimeUnit</unitType>
   <attributeItemSupportedBy>
      <targetApplicationSpecName>Pricing</targetApplicationSpecName>
      <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
   </attributeItemSupportedBy>
</simpleAttributeItem>
<simpleAttributeItem>
   <name>SPECIFIC_UNIT</name>
   <description>SPECIFIC_UNIT</description>
   <type>UNIT_VALUE</type>
   <unitType>Occurrence</unitType>
   <attributeItemSupportedBy>
      <targetApplicationSpecName>Pricing</targetApplicationSpecName>
      <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
   </attributeItemSupportedBy>
</simpleAttributeItem>
```

5. Add REQUESTED_UNITS and USED_UNITS attributes as complex attribute items in the file:

```
<complexAttributeItem>
   <name> REQUESTED_UNITS</name>
   <description>REQUESTED_UNITS</description>
   <type>STRUCT</type>
   <attributeItemSupportedBy>
      <targetApplicationSpecName>Pricing</targetApplicationSpecName>
      <targetApplicationSpecName>Convergent Charging</targetApplicationSpecName>
   </attributeItemSupportedBy>
</complexAttributeItem>
<complexAttributeItem>
   <name>USED_UNITS</name>
   <description>USED_UNITS</description>
   <type>STRUCT</type>
   <attributeItemSupportedBy>
      <targetApplicationSpecName>Pricing</targetApplicationSpecName>
      <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
   </attributeItemSupportedBy>
</complexAttributeItem>
```

6. Copy the event attributes you added in step 4 (if applicable) and paste them in both REQUESTED_UNITS and USED_UNITS complex attribute items:

   For example:

Copyright © 2017, 2026, Oracle and/or its affiliates.

```
<complexAttributeItem>
   <name>REQUESTED_UNITS</name>
   <description>REQUESTED_UNITS</description>
   <type>STRUCT</type>
   <attributeItemSupportedBy>
      <targetApplicationSpecName>Pricing</targetApplicationSpecName>
      <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
   </attributeItemSupportedBy>
   <simpleAttributeItem>
      <name>DURATION</name>
      <description>DURATION</description>
      <type>UNIT_VALUE</type>
      <unitType>TimeUnit</unitType>
      <attributeItemSupportedBy>
         <targetApplicationSpecName>Pricing</targetApplicationSpecName>
         <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
      </attributeItemSupportedBy>
   </simpleAttributeItem>
   <simpleAttributeItem>
      <name>SPECIFIC_UNIT</name>
      <description>SPECIFIC_UNIT</description>
      <type>UNIT_VALUE</type>
      <unitType>Occurrence</unitType>
      <attributeItemSupportedBy>
         <targetApplicationSpecName>Pricing</targetApplicationSpecName>
         <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
      </attributeItemSupportedBy>
   </simpleAttributeItem>
</complexAttributeItem>
<complexAttributeItem>
   <name>USED_UNITS</name>
   <description>USED_UNITS</description>
   <type>STRUCT</type>
      <attributeItemSupportedBy>
         <targetApplicationSpecName>Pricing</targetApplicationSpecName>
         <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
      </attributeItemSupportedBy>
   <simpleAttributeItem>
      <name>DURATION</name>
      <description>DURATION</description>
      <type>UNIT_VALUE</type>
      <unitType>TimeUnit</unitType>
      <attributeItemSupportedBy>
         <targetApplicationSpecName>Pricing</targetApplicationSpecName>
         <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
      </attributeItemSupportedBy>
   </simpleAttributeItem>
   <simpleAttributeItem>
      <name>SPECIFIC_UNIT</name>
      <description>SPECIFIC_UNIT</description>
      <type>UNIT_VALUE</type>
      <unitType>Occurrence</unitType>
      <attributeItemSupportedBy>
         <targetApplicationSpecName>Pricing</targetApplicationSpecName>
         <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
      </attributeItemSupportedBy>
   </simpleAttributeItem>
</complexAttributeItem>
```

7. (Optional) Add transient attributes if required for usage charging:

   For example:

Copyright © 2017, 2026, Oracle and/or its affiliates.

```
<simpleAttributeItem>
   <name>CELL_ID</name>
   <description>CELL_ID</description>
   <type>STRING</type>
   <attributeItemSupportedBy>
   <targetApplicationSpecName>Pricing</targetApplicationSpecName>
   <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
   </attributeItemSupportedBy>
   </simpleAttributeItem>
```

8. Add the charging operation types that you intend to use for usage charging and associate them with the group of event attributes that are necessary to perform the charging operation. These can be **Initiate**, **Terminate**, **Update**, **Debit Unit**, **Refund Amount**, **Start Accounting**, **Update Accounting**, **Price Enquiry**, **IUT**, **Refund Unit**, and **Cancel**.

   The syntax for adding charging operation types is as follows:

```
<opTypes>
   <opType>ChargingOperationType</opType>
   <action>Input</action>
   <item>
     <attributeItemName>EventAttribute</attributeItemName>
     <optional>[true|false]</optional>
   </item>
</opTypes>
```

   where:

   - *ChargingOperationType* is the charging operation type to be used. See Table 34-1 for the list of charging operation types that are used in ECE.
   - *EventAttribute* is the event attribute that is used as input for performing the charging operation.

   The following example shows the charge operation type and required attributes to initiate a charging session for the EventCloudLibrary event:

```
<eventAttributeSpec>
   <name>EventDelayedSession</name>
   <description>EventDelayedSession</description>
   <internalId>7b881d17-de4c-4afd-bb88-71f4dcfc7a67</internalId>
   <priceListName>Default</priceListName>
   <obsolete>false</obsolete>
   <attributeSpecSupportedBy>
      <targetApplicationSpecName>Pricing</targetApplicationSpecName>
      <targetApplicationSpecName>Billing</targetApplicationSpecName>
      <targetApplicationSpecName>Convergent Charging</targetApplicationSpecName>
   </attributeSpecSupportedBy>
   <status>ENABLED</status>
   <className>/event/delayed/session</className>
   <eventType>USAGE_POSTPAID</eventType>
   <baseEventSpec>EventDelayed</baseEventSpec>
   <virtualColumn>false</virtualColumn>
   <unit>NONE</unit>
   <opVersion>
      <externalVersion>1.0</externalVersion>
      <overwrite>false</overwrite>
      <opTypes>
         <opType>Terminate</opType>
         <action>INPUT</action>
      </opTypes>
   </opVersion>
</eventAttributeSpec>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

9. (Optional) Update the charging operation type version number. See "About Charging Operation Type Versions" for more information.
10. Define the network mapping for the events used for charging by doing the following:
    a. Search for the event attribute that you want to map to the network attribute.
    b. Do one of these:
       - If you have a 4G network, add the **networkAttributeItem** entry:

         <networkAttributeItem>*NetworkAttribute***</networkAttributeItem>**
       - If you have a 5G network, add the **networkAttributeItem5g** entry:

         <networkAttributeItem5g>*NetworkAttribute***</networkAttributeItem5g>**

       where *NetworkAttribute* is the attribute of requests received from the network.

    The following example shows the mapping for a 5G network and the EventCloudLibrary event:

```
      <simpleAttributeItem>
         <name>MY_CALLED_ID</name>
         <description>MY_CALLED_ID</description>
         <type>STRING</type>
         <persistedName5g>my_called_id_c</persistedName5g>
         <attributeItemSupportedBy>
           <targetApplicationSpecName>Pricing</targetApplicationSpecName>
           <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
         </attributeItemSupportedBy>
                        <networkAttributeItem5g>Service-Information.PS-
Information.Called-Station-Id</networkAttributeItem5g>
      </simpleAttributeItem>
...
   <complexAttributeItem>
      <name> REQUESTED_UNITS </name>
      <description>REQUESTED_UNITS</description>
      <type>STRUCT</type>
      <attributeItemSupportedBy>
         <targetApplicationSpecName>Pricing</targetApplicationSpecName>
         <targetApplicationSpecName>Convergent Charging</targetApplicationSpecName>
      </attributeItemSupportedBy>
      <simpleAttributeItem>
         <name>DURATION</name>
         <description>DURATION</description>
         <type>UNIT_VALUE</type>
         <unitType>TimeUnit</unitType>
         <attributeItemSupportedBy>
            <targetApplicationSpecName>Pricing</targetApplicationSpecName>
            <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
         </attributeItemSupportedBy>
         <networkAttributeItem5g>Multiple-Services-Credit-Control.Requested-Service-
Unit.CC-Time</networkAttributeItem5g>
      </simpleAttributeItem>
      <simpleAttributeItem>
         <name>SPECIFIC_UNIT</name>
         <description>SPECIFIC_UNIT</description>
         <type>UNIT_VALUE</type>
         <unitType>Occurrence</unitType>
         <attributeItemSupportedBy>
            <targetApplicationSpecName>Pricing</targetApplicationSpecName>
            <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
         </attributeItemSupportedBy>
         <networkAttributeItem5g>Multiple-Services-Credit-Control.Requested-Service-
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
Unit.CC-Service-Specific-Units</networkAttributeItem5g>
      </simpleAttributeItem>
   </complexAttributeItem>
   <complexAttributeItem>
      <name>USED_UNITS</name>
      <description>USED_UNITS</description>
      <type>STRUCT</type>
         <attributeItemSupportedBy>
            <targetApplicationSpecName>Pricing</targetApplicationSpecName>
            <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
         </attributeItemSupportedBy>
      <simpleAttributeItem>
         <name>DURATION</name>
         <description>DURATION</description>
         <type>UNIT_VALUE</type>
         <unitType>TimeUnit</unitType>
         <attributeItemSupportedBy>
            <targetApplicationSpecName>Pricing</targetApplicationSpecName>
            <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
         </attributeItemSupportedBy>
         <networkAttributeItem5g>Multiple-Services-Credit-Control.Used-Service-
Unit.CC-Time</networkAttributeItem5g>
      </simpleAttributeItem>
      <simpleAttributeItem>
         <name>SPECIFIC_UNIT</name>
         <description>SPECIFIC_UNIT</description>
         <type>UNIT_VALUE</type>
         <unitType>Occurrence</unitType>
         <attributeItemSupportedBy>
            <targetApplicationSpecName>Pricing</targetApplicationSpecName>
            <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
         </attributeItemSupportedBy>
         <networkAttributeItem5g>Multiple-Services-Credit-Control.Used-Service-
Unit.CC-Service-Specific-Units</networkAttributeItem5g>
      </simpleAttributeItem>
   </complexAttributeItem>
   <complexAttributeItem>
      <name>USAGE_INFO</name>
      <description>USAGE_INFO</description>
      <type>STRUCT</type>
      <persistedName5g>cloudlibrary_usage_info_t</persistedName5g>
      <simpleAttributeItem>
         <name>SERVICE_PROVIDER</name>
         <description>SERVICE_PROVIDER_C</description>
         <type>STRING</type>
         <persistedName5g>service_provider_c</persistedName5g>
         <attributeItemSupportedBy>
            <targetApplicationSpecName>Pricing</targetApplicationSpecName>
            <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
         </attributeItemSupportedBy>
         <networkAttributeItem5g>Multiple-Services-Credit-Control.Used-Service-
Unit.CC-Time</networkAttributeItem5g>
         <length>60</length>
      </simpleAttributeItem>
      <simpleAttributeItem>
         <name>SERVICE_STATUS</name>
         <description>SERVICE_STATUS_C</description>
         <type>STRING</type>
         <persistedName5g>service_status_c</persistedName5g>
         <attributeItemSupportedBy>
            <targetApplicationSpecName>Pricing</targetApplicationSpecName>
            <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
            </attributeItemSupportedBy>
            <networkAttributeItem5g>Multiple-Services-Credit-Control.Used-Service-
Unit.CC-Time</networkAttributeItem5g>
            <length>60</length>
         </simpleAttributeItem>
         <simpleAttributeItem>
            <name>SERVICE_TYPE</name>
            <description>SERVICE_TYPE_C</description>
            <type>STRING</type>
            <persistedName5g>service_type_c</persistedName5g>
            <attributeItemSupportedBy>
               <targetApplicationSpecName>Pricing</targetApplicationSpecName>
               <targetApplicationSpecName>ConvergentCharging</targetApplicationSpecName>
            </attributeItemSupportedBy>
            <networkAttributeItem5g>Multiple-Services-Credit-Control.Used-Service-
Unit.CC-Time</networkAttributeItem5g>
            <length>60</length>
         </simpleAttributeItem>
      </complexAttributeItem>
```

11. Save and close the file.
12. Import the XML file into PDC by running the following command:

    ImportExportPricing -import -metadata *FileOrDirectoryName* **-ow**

    For example:

    ```
    ImportExportPricing -import -metadata EventCloudLibrary_export_metadata.xml -ow
    ```

    The EventCloudLibrary event definition is loaded into PDC. The ECE Pricing Updater publishes the enriched event definition to ECE.
13. Update the version number in your mediation specification file to associate the usage request builder with the event definition to which it applies. See the discussion about editing the mediation specification file in *ECE Implementing Charging* for more information.

You may need to restart your charging client to receive usage requests that adhere to the enriched event definition.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Note

The **ECEEnrichment.xml** file is only used if there are no BCC **eventAttributeSpec** objects in PDC for adding ECE fields into BRM event fields. Once you have added BCC **EventAttribute** objects in PDC, you must enhance the **EventAttributeSpec** using the **ImportExportPricing** utility.

For example, if the fields **Pricing** and **Convergent Charging** are used in ECE only, the **<attributeItemSupportedBy>** contains

the following fields:

```
<targetApplicationSpecName>Pricing</targetApplicationSpecName>
<targetApplicationSpecName>Convergent Charging</
targetApplicationSpecName>
```

If the **<attributeItemSupportedBy>** contains a field that is not defined in ECE, for example:

```
<targetApplicationSpecName>Billing</
targetApplicationSpecName>
<targetApplicationSpecName>Realtime Charging</
targetApplicationSpecName>
```

SyncPDC assumes that this field is defined in BRM. If the field does not exist for the event in BRM, **SyncPDC** will remove it from the BCC **EventAttributeSpec**.

To ensure the attributes are synced to PDC when you run **SyncPDC**, add the event customizations to **ECEEventEnrichmentSpec.xml** file present in *<PDC_BRM_HOME>***/apps/syncpdc**. To do so, set the following fields in **SyncPDCConfiguration.xml** file:

```
<ECESync>TRUE</ECESync>
<skipBREMigration>true</skipBREMigration>
```

The customizations should be added in **ECEEventEnrichmentSpec.xml** file and PDC to get the Enrichment XML attributes.

Editing Event Definitions

To edit an event definition, update the event definition in PDC by using the **ImportExportPricing** utility. When you edit an event definition, ensure that you use the same event type and service type, and update the charging operation type version as appropriate. See "About Charging Operation Type Versions" for more information.

When you create your usage request builder, you must associate it with the service, event type, and charging operation type version in the event definition to which it applies. You may need to restart your charging client to receive usage requests that adhere to the enriched event definition.

Copyright © 2017, 2026, Oracle and/or its affiliates.

35

Using the PDC Web Service

Learn how to use the Oracle Communications Pricing Design Center (PDC) Web Service.

Topics in this document:

- About the PDC Web Service API
- About Using the PDC Web Service
- Creating and Modifying Pricing Components
- Exporting Components

About the PDC Web Service API

The PDC Web service application programming interface (API) allows other Oracle applications as well as third-party applications to interact with PDC. Use the PDC Web service API to:

- Create, modify, and export pricing components.
- Create, modify, and export setup components.
- Create and export metadata and custom data.

To use the PDC Web service API, the external application must call the PDC Web service API through a custom client application. You can use any language that supports Web services to access the API, for example, Java.

About Using the PDC Web Service

To use the PDC Web service, you must:

- Ensure that the SSL port for the WebLogic Server domain is enabled. See "Configuring SSL for the WebLogic Server Domain" in *PDC Installation Guide* for more information.
- Ensure that the following system properties are set in WebLogic Server:
  - weblogic.security.TrustKeyStore
  - weblogic.security.CustomTrustKeyStoreFileName
  - weblogic.security.CustomTrustKeyStorePassPhrase
  - weblogic.security.CustomTrustKeyStoreType

  See the discussion about setting the system properties in Oracle WebLogic Remote Console Online Help for more information.
- Ensure that the following BindingProvider properties are set in WebLogic Server:
  - BindingProvider.USERNAME_PROPERTY
  - BindingProvider.PASSWORD_PROPERTY

  See the discussion about setting the BindingProvider properties in Oracle WebLogic Remote Console Online Help for more information.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Creating and Modifying Pricing Components

To create or modify components with the PDC Web service:

- Create the XML file containing the components. See "About Creating the XML Files" for more information.
- Use the PDC Web service to create or modify the components defined in the XML file. The PDC APIs are exposed as Web service operations through the **PricingGateway.wsdl** file. This file defines the Web service that can be called as well as the attributes required to call a specific operation.

You can use the PDC Web services for the tasks shown in Table 35-1.

Table 35-1 Create and Modify Components

Task	Operation
Create pricing components.	CreatePricing
Modify promoted pricing components.	ModifyPricing
Create pricing components and publish the components to the Oracle Communications Billing and Revenue Management (BRM) database.	CreatePricingAndSubmit
Modify promoted pricing components and publish the components to the BRM database.	ModifyPricingAndSubmit
Create setup components.	createBusinessConfig
Create setup components and publish the setup components to a billing system such as BRM.	createBusinessConfigAndSubmit
Modify setup components.	modifyBusinessConfig
Modify setup components and publish the setup components to a billing system such as BRM.	modifyBusinessConfigAndSubmit
Create metadata.	createMetadata
Create custom data.	createCustomFields

About Creating the XML Files

You need to create an XML file containing the components you want to use. You can create:

- One consolidated XML file containing all the components
- Separate XML files for each type of component (for example, charge offers, discount offers, packages, and so on)

The XML files that you create must conform to the format detailed in the XSD files for pricing components.

The PDC Web service uses the **PricingGateway.xsd** file for creating or modifying components in PDC. The XSD files describe the structure of the XML document. The XML file you create must comply with the structure defined in the XSD. The **PricingGateway.xsd** file is available at:

`http://`*`hostName`*`:`*`sslPortNumber`*`/pdc/PricingGatewayPort?xsd=1`

where:

- *hostName* is the host name of the machine on which PDC is deployed.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- *sslPortNumber* is the SSL port number of the domain on which PDC is deployed.

Sample Code to Call the PDC Web Service

Use the following sample code to call the PDC Web service:

```
PricingGatewayPortType pricingGatewayPortType
=(PricingGatewayPortType)Service.getPort(new
QName("urn:PricingGateWay","PricingGatewayPort"),
PricingGatewayPortType.class);

Map<String, Object> rc = ((BindingProvider)pricingGatewayPortType).getRequestContext();
rc. put(BindingProvider.ENDPOINT_ADDRESS_PROPERTY, http://hostName:sslPortNumber/pdc/
PricingGatewayPort?WSDL));
rc.put(BindingProvider.USERNAME_PROPERTY,pdcUserName);
rc.put(BindingProvider.PASSWORD_PROPERTY,pdcUserPassword);

UserContextType userContext = new UserContextType();
userContext.setUserid("pdcUserName");

PricingInputXMLType pricingInputXMLType = new PricingInputXMLType();
pricingInputXMLType.setUserContext(userContext);
ByteArrayOutputStream bs = new ByteArrayOutputStream();
ZipOutputStream out = new ZipOutputStream(bs);
        FileInputStream fl = null;
        try {
            fl = new FileInputStream(new File(pricing.xml));
            // Set the compression ratio
            out.setLevel(Deflater.BEST_COMPRESSION);
            ZipEntry ze = new ZipEntry(PDC);
            out.putNextEntry(ze);
            byte[] data = new byte[BUFFER_SIZE];            int count = 0;
            BufferedInputStream in = new BufferedInputStream(fl);
            while ((count = in.read(data, 0, BUFFER_SIZE)) != -1){
                out.write(data, 0, count);
            }
        } catch (Exception e) {
            throw e;
        } finally {
            if (fl != null) {
                fl.close();
            }
            if (bs != null) {
                bs.flush();
            }
            if (out != null) {
                out.flush();
                out.close();
            }
        }
        byte[] bytesToRet = null;
        if (bs != null) {
            bytesToRet = bs.toByteArray();
        }
pricingInputXMLType.setXmlBinaryString(bytesToRet);

PDCResponseType pDCResponseType =
pricingGatewayPortType.createPricingAndSubmit(pricingInputXMLType);
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

CreatePricing

This Web service operation validates the input XML by comparing the XML fields and values against the values in the **PricingGateway.xsd** file and the rules for each type of pricing component. If the validation is successful, it retrieves the data from the XML file and creates the pricing components in PDC.

The **CreatePricing** operation does not publish the pricing components to the BRM database.

Syntax

```
public oracle.communications.brm.pdc.server.service.types.PDCResponseType
createPricing(oracle.communications.brm.pdc.server.service.types.PricingInputXMLType
param) throws oracle.communications.brm.pdc.server.service.PricingExceptionResponse;
```

ModifyPricing

This Web service operation validates the input XML by comparing the XML fields and values against the values in the **PricingGateway.xsd** file and the rules for each type of pricing component. If the validation is successful, it retrieves the data from the XML file and updates the existing pricing components in PDC.

The **ModifyPricing** operation does not publish the pricing components to the BRM database.

Syntax

```
public oracle.communications.brm.pdc.server.service.types.PDCResponseType
modifyPricing(oracle.communications.brm.pdc.server.service.types.PricingInputXMLType
param) throws oracle.communications.brm.pdc.server.service.PricingExceptionResponse;
```

CreatePricingAndSubmit

This Web service operation validates the input XML by comparing the XML fields and values against the values in the **PricingGateway.xsd** file and the rules for each type of pricing component. If the validation is successful, it retrieves the data from the XML file, creates the pricing components in PDC, and publishes the pricing components to the BRM and ECE databases.

Syntax

```
public oracle.communications.brm.pdc.server.service.types.PDCResponseType
createPricingAndSubmit(oracle.communications.brm.pdc.server.service.types.PricingInputXML
Type param) throws oracle.communications.brm.pdc.server.service.PricingExceptionResponse;
```

ModifyPricingAndSubmit

This Web service operation validates the input XML by comparing the XML fields and values against the values in the **PricingGateway.xsd** file and the rules for each type of pricing component. If the validation is successful, it retrieves the data from the XML file, updates the existing pricing components in PDC, and publishes the pricing components to the BRM and ECE databases.

Syntax

```
public oracle.communications.brm.pdc.server.service.types.PDCResponseType
modifyPricingAndSubmit(oracle.communications.brm.pdc.server.service.types.PricingInputXML
Type param) throws oracle.communications.brm.pdc.server.service.PricingExceptionResponse;
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

createBusinessConfig

This Web service operation validates the input XML by comparing the XML fields and values against the values in the **PricingGateway.xsd** file and the rules for each type of business config component. If the validation is successful, it retrieves the data from the XML file and creates business config components in PDC.

The **createBusinessConfig** operation does not publish the business config components to the BRM database.

Syntax

```
public oracle.communications.brm.pdc.server.service.types.PDCResponseType
createBusinessConfig(oracle.communications.brm.pdc.server.service.types.PricingInputXMLTy
pe param) throws oracle.communications.brm.pdc.server.service.PricingExceptionResponse;
```

createBusinessConfigAndSubmit

This Web service operation validates the input XML by comparing the XML fields and values against the values in the **PricingGateway.xsd** file and the rules for each type of business config component. If the validation is successful, it retrieves the data from the XML file, creates business config components in PDC, and publishes the business config components to the BRM database.

Syntax

```
public oracle.communications.brm.pdc.server.service.types.PDCResponseType
createBusinessConfigAndSubmit(oracle.communications.brm.pdc.server.service.types.PricingI
nputXMLType param) throws
oracle.communications.brm.pdc.server.service.PricingExceptionResponse;
```

modifyBusinessConfig

This Web service operation validates the input XML by comparing the XML fields and values against the values in the **PricingGateway.xsd** file and the rules for each type of business config component. If the validation is successful, it retrieves the data from the XML file and updates the existing business config components in PDC.

The **modifyBusinessConfig** operation does not publish the business config components to the BRM database.

Syntax

```
public oracle.communications.brm.pdc.server.service.types.PDCResponseType
modifyBusinessConfig(oracle.communications.brm.pdc.server.service.types.PricingInputXMLTy
pe param) throws oracle.communications.brm.pdc.server.service.PricingExceptionResponse;
```

modifyBusinessConfigAndSubmit

This Web service operation validates the input XML by comparing the XML fields and values against the values in the **PricingGateway.xsd** file and the rules for each type of business config component. If the validation is successful, it retrieves the data from the XML file and updates the existing business config components in PDC, and publishes the business config components to the BRM database.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Syntax

```
public oracle.communications.brm.pdc.server.service.types.PDCResponseType
modifyBusinessConfigAndSubmit(oracle.communications.brm.pdc.server.service.types.PricingI
nputXMLType param) throws
oracle.communications.brm.pdc.server.service.PricingExceptionResponse;
```

createMetadata

This Web service operation validates the input XML by comparing the XML fields and values against the values in the **PricingGateway.xsd** file and the rules for each type of metadata component. If the validation is successful, it retrieves the data from the XML file and creates metadata components in PDC.

The **createMetadata** operation does not publish the metadata components to the BRM database.

Syntax:

```
public oracle.communications.brm.pdc.server.service.types.PDCResponseType
createMetadata(oracle.communications.brm.pdc.server.service.types.PricingInputXMLType
param) throws oracle.communications.brm.pdc.server.service.PricingExceptionResponse;
```

createCustomFields

This Web service operation validates the input XML by comparing the XML fields and values against the values in the **PricingGateway.xsd** file and the rules for each type of custom fields. If the validation is successful, it retrieves the data from the XML file and creates custom fields in PDC.

The **createCustomFields** operation does not publish the custom fields to the BRM database.

Syntax

```
public oracle.communications.brm.pdc.server.service.types.PDCResponseType
createCustomFields(oracle.communications.brm.pdc.server.service.types.PricingInputXMLType
 param) throws oracle.communications.brm.pdc.server.service.PricingExceptionResponse;
```

Exporting Components

You can use the PDC Web services for the tasks shown in Table 35-2.

Table 35-2 Export Tasks

Task	Operation
Export pricing components.	getPricingObjects
Export setup components.	getBusinessConfigObjects
Export metadata components.	getMetadataObjects
Export BRM components.	getBRMObjects
Export custom fields.	getCustomFields

PDC Web services use the **RetrieveInputXMLType** object as input for exporting the PDC components. You can create the **RetrieveInputXMLType** object.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- Using the Setter Method. See "Creating RetrieveInputXMLType Object Using Setter Method".
- Using the XML. See "Creating RetrieveInputXMLType Object Using XML".

Creating RetrieveInputXMLType Object Using Setter Method

You can create the **RetrieveInputXMLType** object using setter methods (for example, pojos) with the required parameters to export components from PDC system.

For example, to export pricing components you can create the object as follows:

```
getPricingObjects() {
 RetrieveResponseType response = new RetrieveResponseType();
 RetrieveInputXMLType request = new RetrieveInputXMLType();
 ObjectType objectType = new ObjectType();
 WSPricingObject pricingObject = new WSPricingObject();
 pricingObject.getPricingObjectType().add(WSPricingObjectType.ALTERATION_EXCLUSION);
 objectType.setPricingObject(pricingObject);
 request.setObjectType(objectType);
 request.setAllReferences(false);
 request.setIncludeFailedObjects(false);
 request.setObsolete(false);
 request.setReferences(false);
 UserContextType userContext = new UserContextType();
 userContext.setUserid("pdcuser");
 request.setUserContext(userContext);
 response = pricingGatewayPortType.getPricingObjects(request);
 return response;
}
```

Creating RetrieveInputXMLType Object Using XML

The PDC Web services use the **RetrievePricingGateway.xsd** file for exporting components from PDC. The XSD file describes the structure of the XML file. The XML file you create must comply with the structure defined in the XSD. The **RetrievePricingGateway.xsd** file is available at:

http://*hostName*:*sslPortNumber*/pdc/PricingGatewayPort?xsd=2**

where:

- *hostName* is the host name of the machine on which PDC is deployed.
- *sslPortNumber* is the SSL port number of the domain on which PDC is deployed.

You need to manually set the required parameters in the XML file for indicating the objects that you want to export from the PDC system. The **RetrieveInputXMLType** object is then created by parsing this XML.

Parameters for Exporting PDC Components

Table 35-3 describes the parameters used for exporting PDC components.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 35-3 Parameters for Exporting PDC Components

Parameter	Description
objectType	Specifies the object type to export. The valid values are **pricingObject**, **businessConfigObject**, **brmObject**, and **metadataObject**. You can specify the pricing object type for exporting. For example, ALTERATION_RATE_PLAN, USC_MAP and so on. If pricing object type is not specified, all the pricing components are considered for exporting. This parameter is also applicable for setup components, metadata, and cross-reference data.
references	Specifies whether to export data including references.
allReferences	Specifies whether to export data including all references. This is applicable only when exporting pricing objects. When **allReferences** is set to **true** for pricing objects, the response is written separately into **pricingObjectResponse**, **configObjectResponse**, and **metadataObjectResponse** fields. If the export **allReferences** is not set for pricing objects, the pricing object responses are written into **response** field.
obsolete	Specifies whether to export obsolete data.
includeFailedObjects	Specifies whether to export data including failed objects.
modifiedAfterDate	Specifies whether to export data based on modified date. **ModifiedAfterDate** supports the following formats: • *yyyy-mm-dd* Example: 2010-01-05 • *yyyy-mm-dd'T'hh:mm:ss* Example: 2010-01-05T23:59:59 • *yyyy-mm-dd'T'hh:mm:ssz* Example: 2010-01-05T19:05:09GMT+05:30
modifiedByUser	Specifies whether to export data based on the user who modified the data.
objectName	Specifies whether to export data based on object name.
UserContext	Specifies whether to export data for a specific user.
productSpecName	Specifies whether to export data based on product specification. This is applicable only when exporting pricing components.

Web Service Response

The PDC Web service operation first validates the input XML by comparing the XML fields and its values against the values in the **RetrievePricingGateway.xsd** file. If the validation is successful, it retrieves the objects from PDC and returns the data in **RetrieveResponseType** object.

Here is the structure of the **RetrieveResponseType** object:

```
<xsd:complexType name="RetrieveResponseType">
 <xsd:annotation>
  <xsd:documentation>Get Object Webservice Response</xsd:documentation>
 </xsd:annotation>
 <xsd:sequence>
  <xsd:element name="status" type="xsd:string"/>
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
  <xsd:element name="errors" type="xsd:string" maxOccurs="unbounded" minOccurs="0"/>
  <xsd:element name="response" type="xsd:base64Binary"/>
  <xsd:element name="pricingObjectResponse" type="xsd:base64Binary"/>
  <xsd:element name="configObjectResponse" type="xsd:base64Binary"/>
  <xsd:element name="metadataObjectResponse" type="xsd:base64Binary"/>
 </xsd:sequence>
</xsd:complexType>
```

The **RetrieveResponseType** object contains the status of operation and the exported object fields in the XML binary format based on the input provided in **RetrieveInputXml** file. The status field value is either SUCCEEDED or FAILED. The **RetrieveResponseType** object also capture errors occurred during the operation.

Processing the Response from Export/GET API

When an export or GET operation is successful, you receive a response with the requested object data. You get the response as follows:

```
response = pricingGatewayPortType.getPricingObjects(request);
```

You can process the response object by using the following code in your application:

```
processResponse(RetrieveResponseType response) {
byte[] zippedBinaryStr = response.getResponse();
for (int i = 0; i < response.getErrors().size(); i++) {
  System.out.println("Cause : \n" + response.getErrors().get(i)+ "\n");
}
String status = response.getStatus(); //status values can be either SUCCEEDED or FAILED
}
```

Optionally, you can convert the response into the XML format by using the following code in your application:

```
private static void writeToFile(byte[] zippedBinaryStr, String filename) throws
Exception {
        try {
            FileOutputStream fpStream = new FileOutputStream(filename);
            BufferedOutputStream bufferedStream = new BufferedOutputStream(fpStream);
            extractZippedContentAndWriteXMLStream(zippedBinaryStr, bufferedStream);
            bufferedStream.flush();
            bufferedStream.close();
            fpStream.flush();
            fpStream.close();
        } catch (IOException e) {
            System.out.println("Exception caught when writing data to file");
            throw e;
        }
    }
   private static void extractZippedContentAndWriteXMLStream(byte[] binaryXMLString,
BufferedOutputStream bufferedFileStream) throws Exception {
        try (ByteArrayInputStream byteStrm = new ByteArrayInputStream(binaryXMLString);
             ZipInputStream in = new ZipInputStream((byteStrm))) {
             ZipEntry entry = in.getNextEntry();
            //Transfer bytes from the ZIP file to the output file
            if (entry != null && in.available() > 0) {
                byte[] buf = new byte[BUFFER_SIZE];
                int len = in.read(buf);
                while (len > 0) {
                     bufferedFileStream.write(buf, 0, len);
                    len = in.read(buf);
                }
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

```
                bufferedFileStream.flush();
            }
        } catch (Exception e) {
            throw e;
        }
    }
```

where *filename* is the path to the XML file for storing the output.

getPricingObjects

This Web service operation validates the input XML by comparing the XML fields and values against the values in the **RetrievePricingGateway.xsd** file. If the validation is successful, it retrieves the pricing objects from PDC and returns the data in **RetrieveResponseType** object.

Syntax

```
public oracle.communications.brm.pdc.server.service.types.RetrieveResponseType
getPricingObjects(oracle.communications.brm.pdc.server.service.types.RetrieveInputXMLType
 param) throws Exception;
```

getMetadataObjects

This Web service operation validates the input XML by comparing the XML fields and values against the values in the **RetrievePricingGateway.xsd** file. If the validation is successful, it retrieves the metadata object from PDC and returns the data in **RetrieveResponseType** object.

Syntax

```
public oracle.communications.brm.pdc.server.service.types.RetrieveResponseType
getMetadataObjects(oracle.communications.brm.pdc.server.service.types.RetrieveInputXMLTyp
e param) throws Exception;
```

getCustomFields

This Web service operation validates the input XML by comparing the XML fields and values against the values in the **RetrievePricingGateway.xsd** file. If the validation is successful, it retrieves the custom fields from PDC and returns the data in **RetrieveResponseType** object.

Syntax

```
public oracle.communications.brm.pdc.server.service.types.RetrieveResponseType
getCustomFields(oracle.communications.brm.pdc.server.service.types.RetrieveInputXMLType
param) throws Exception;
```

getBusinessConfigObjects

This Web service operation validates the input XML by comparing the XML fields and values against the values in the **RetrievePricingGateway.xsd** file. If the validation is successful, it retrieves the business config from PDC and returns the data in **RetrieveResponseType** object.

Syntax

```
public oracle.communications.brm.pdc.server.service.types.RetrieveResponseType
getBusinessConfigObjects(oracle.communications.brm.pdc.server.service.types.RetrieveInput
XMLType param) throws Exception;
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

getBRMObjects

This Web service operation validates the input XML by comparing the XML fields and values against the values in the **RetrievePricingGateway.xsd** file. If the validation is successful, it retrieves the BRM object from PDC and returns the data in **RetrieveResponseType** object.

Syntax

```
public oracle.communications.brm.pdc.server.service.types.RetrieveResponseType
getBRMObjects(oracle.communications.brm.pdc.server.service.types.RetrieveInputXMLType
param) throws Exception;
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

36
Working with Profiles

Learn about the profiles used by Pricing Design Center (PDC) as well as how to create custom profiles for Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

- About Pricing Profiles
- About Default Pricing Profiles
- About Customizing Pricing Profiles
- Setting Up Custom Pricing Profiles

About Pricing Profiles

A *pricing profile* specifies the pricing features that are displayed in the PDC UI. You use pricing profiles to adapt the UI to your business needs and to simplify the user experience. For example, if you do not use time to determine how much to charge for the use of your services, you can set up a pricing profile that does not include the time feature.

Pricing profiles are used with pricing, charges, discounts, chargeshares, selectors, and time models. You select a pricing profile for all those components except pricing when creating them. (PDC selects a profile for pricing that matches the pricing profile of the component that contains the pricing, such as a charge.)

A set of default pricing profiles is included with PDC (see "About Default Pricing Profiles"). You can also create custom pricing profiles (see "About Customizing Pricing Profiles").

About Default Pricing Profiles

During Oracle Communications Pricing Design Center (PDC) installation, depending on whether you selected to support Oracle Communications Elastic Charging Engine (ECE) or not, a set of default pricing profiles is loaded in the XML format. The XML files containing the default pricing profiles are located in the *PDC_home***/apps/Samples/Examples** directory, where *PDC_home* is the directory in which the PDC software is installed.

Table 36-1 lists the default pricing profiles that are loaded during PDC installation if ECE is selected. These pricing profiles are supported by ECE rating engine.

Table 36-1 Default Pricing Profiles That Are Loaded When ECE Is Selected

Profile Name	Description
Billing-time Discount	Contains all the features required to configure billing-time discounts in BRM (see Table 36-5). This profile can be used with a discount or a discount selector.
Convergent Usage	Contains all the pricing features supported by ECE (see Table 36-8). This profile can be used with a usage charge, time model, or generic selector.
Standard Chargesharing	Contains all the chargeshare features supported by ECE (see Table 36-10). This profile can be used with a chargeshare.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 36-1 (Cont.) Default Pricing Profiles That Are Loaded When ECE Is Selected

Profile Name	Description
Standard Discount	Contains all the discount features supported by ECE (see Table 36-9). This profile can be used with a discount or a discount selector.
Subscription ChargeSharing	Contains all the chargeshare features supported by the BRM offline rating engine (see Table 36-6). This profile can be used with a chargeshare.
Subscription Discount	Contains all the discount features supported by the BRM offline rating engine (see Table 36-5). This profile can be used with a discount or a discount selector.

See "About Customizing Pricing Profiles" for information about modifying the default pricing profiles.

About Customizing Pricing Profiles

Typically, you create a custom pricing profile by copying one of the default pricing profiles and then removing features that you do not need from the copy.

Note

- In profile XML files, pricing features are defined in **<capability>** and **<subCapability>** elements.
- You cannot modify the Subscription pricing profile.

For example, if your business does not charge for services by time, you could remove the **TimeModeling** capability from the Online Usage default pricing profile.

Note

If a capability is mandatory in the target engine profile associated with a pricing profile, the capability must be included in the pricing profile. Do not remove it. See "About the PDC Target Engine Profiles" for a list of capabilities supported by each PDC target engine profile. The list identifies the mandatory capabilities.

If two or more mandatory capabilities are mutually exclusive, you can remove all but one of them from a custom profile. If you include more than one, you must also include the exclusive constraint (**<exclusiveConstraint>**) between them. See the default pricing profiles for examples.

The following rules apply for *subcapabilities*:

- If only one subcapability exists for a particular capability in a target engine profile, the pricing profile must include that subcapability. Do not remove it.
- If multiple subcapabilities exist for a particular capability in a target engine profile, at least one of those subcapabilities must be included in the pricing profile. The custom pricing profile does not, however, have to use the same subcapability as the default pricing profile.

Copyright © 2017, 2026, Oracle and/or its affiliates.

For example, suppose a target engine profile includes Capability X, which has Subcapabilities A, B, C, and D, and a default pricing profile uses Capability X with Subcapabilities A and B. If you customize a copy of that default pricing profile, the copy could include Capability X with Subcapability D. In this case, the custom pricing profile does not use the same subcapabilities as the default pricing profile.

- If you *add* capabilities to a custom pricing profile, ensure the profile contains capabilities for only one pricing type. See "About Pricing Types" for more information.
- Only capabilities that are supported by all the target engine profiles associated with a pricing profile can be added to a custom pricing profile. See "About the PDC Target Engine Profiles" for a list of capabilities supported by each PDC target engine profile. See "About Associating Custom Pricing Profiles with Target Engine Profiles" for information about associating pricing profiles with target engine profiles.

> ⚠ **Caution**
>
> After pricing components are associated with a pricing profile, you should not modify that pricing profile. Doing so might invalidate the pricing components that reference the profile.

See "Setting Up Custom Pricing Profiles" for instructions on how to create custom profiles.

About Pricing Types

Each pricing profile can support only one pricing type. A *pricing type* performs the following functions:

- Indicates the type of pricing, such as a usage charge or a discount, that can be configured when the pricing profile is associated with a pricing component
- For custom profiles, determines the set of capabilities that are available for inclusion in the profile

In each pricing profile's XML file, the pricing type is specified in the **<pricingTypeValue>** element.

Table 36-2 shows the pricing types supported by the default pricing profiles.

Table 36-2 Pricing Types Supported by the default Pricing Profiles

Default Pricing Profile	Supported Pricing Type
Online Usage	USAGE_CHARGE
Offline Usage	USAGE_CHARGE
Common Usage	USAGE_CHARGE
(For BRM) Standard Discount	DISCOUNT
Billing-time Discount	RECURRING_DISCOUNT
ChargeShare	CHARGE_SHARE
Convergent Usage	USAGE_CHARGE
(For ECE) Standard Discount	DISCOUNT
Standard Chargesharing	CHARGE_SHARE

Copyright © 2017, 2026, Oracle and/or its affiliates.

About the PDC Target Engine Profiles

PDC uses the following target engine profiles:

> **Note**
>
> During PDC installation, all the target engine profiles are loaded irrespective of whether ECE is supported or not.

- **BRM online rating:** Contains all the pricing capabilities (features) supported by the BRM online rating engine. See "BRM Online Rating Target Engine Profile" for a list of the capabilities in this profile.
- **BRM offline rating:** Contains all the pricing capabilities supported by the BRM offline rating engine. See "BRM Offline Rating Target Engine Profile" for a list of the capabilities in this profile.
- **Subscription:** Contains capabilities required to create components supported by the BRM subscription engine, such as recurring charges, one-time charges, rollovers, folds, and bundles. See "Subscription Engine Profile" for a list of the capabilities in this profile.
- **ECE rating:** Contains all the pricing capabilities supported by ECE. See "ECE Rating Target Engine Profile" for a list of the capabilities in this profile.
- **ECE subscription:** Contains capabilities to support product offerings. See "ECE Subscription Engine Profile" for a list of the capabilities in this profile.

> **Note**
>
> - If a target engine capability is identified as mandatory in the following tables, it must be included in any pricing profile that is associated with the target engine profile.
> - If two or more mandatory capabilities are mutually exclusive, you can remove all but one of them from a custom profile. If you include more than one, you must also include the exclusive constraint (**<exclusiveConstraint>**) between them. See the default pricing profiles for examples.
> - You cannot customize a target engine profile.

BRM Online Rating Target Engine Profile

The name of the BRM online rating target engine profile is RRE_RATING.

The pricing type of all the capabilities in this profile is USAGE_CHARGE (see "About Pricing Types").

Table 36-3 lists the capabilities contained in this target engine profile.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 36-3 Capabilities In the BRM Online Rating Target Engine Profile

Capability	Description
RateplanSelector	Enables you to use a charge selector when configuring a charge. Cannot be used with the EnhancedZoneModeling capability.
ChargeRateplan	(Mandatory) Enables you to configure charges for a charge offer.
SubscriberCurrency	(Mandatory) Enables you to specify a currency for a charge. **Multicurrency:** This subcapability enables you to select one or more currencies in the **Currency** field in the Create Charge dialog box .
MultiRumGraph	Enables you to select one or more RUMs in the **Measured By** field in the Create Charge dialog box.
AbsoluteRelativeDateTimeRange	Enables users to specify fixed and relative date ranges when configuring a charge.
EnhancedZoneModeling	Enables you to use zone models and Usage Scenario (USC) selectors when configuring a charge. **USCSelector:** This subcapability enables you to use USC selectors when configuring a charge. Cannot be used with the RateplanSelector capability.
TimeModeling	Enables you to use time models when configuring a usage charge. The following subcapabilities specify which attributes of a time model can be used: • **MonthOfYear** • **DayOfWeek** • **TimeOfDay** • **DayOfMonth**
UsageBI	(Mandatory if PriceSelector is not included) Enables you to configure one or more balance impacts for a usage charge. **CreditLimitCheck:** This subcapability enables you to specify whether to continue rating the remaining quantity after the credit limit is exceeded or to search for a charge that impacts a different balance element whose credit limit has not been exceeded and use that charge instead. **StepsBasedOnUsageQuantity:** This subcapability enables you to create price steps based on event quantity. **StepsBasedOnBalanceElement:** This subcapability enables you to create price steps based on balance element quantity. **SingleRum:** This subcapability enables you to select only one ratable usage metric (RUM) in the **Measured By** field in the Create Charge dialog box. Cannot be used with the PriceSelector capability.

BRM Offline Rating Target Engine Profile

The name of the BRM offline rating target engine profile is BRE_RATING.

This profile contains capabilities for the following pricing types:

- USAGE_CHARGE (see Table 36-4)
- DISCOUNT and RECURRING_DISCOUNT (see Table 36-5)
- CHARGE_SHARE (see Table 36-6)

See "About Pricing Types" for more information.

Table 36-4 lists the charging capabilities contained in this target engine profile:

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 36-4 Charging Capabilities in the BRM Offline Rating Target Engine Profile

Capability	Description
ChargeRateplan	(Mandatory) Enables you to configure charges for a charge offer. **MultiRum:** This subcapability enables you to select one or more RUMs in the **Measured By** field in the Create Charge dialog box.
SubscriberCurrency	(Mandatory) Enables you to specify the currency to use in the charge. **SingleCurrency** This subcapability enables you to select only one currency in the **Currency** field in the Create Charge dialog box.
AbsoluteDateRange	Enables you to specify only fixed (not relative) date ranges when configuring a charge.
ZoneModeling	Enables you to use zone models when configuring a charge. Cannot be used with the EnhancedZoneModeling capability.
EnhancedZoneModeling	Enables you to use zone models, USC selectors, and Access Point Name (APN) selectors when configuring a charge. **USCSelector:** This subcapability enables you to use a USC selector when configuring a charge. **APNSelector:** This subcapability enables you to use an APN selector when configuring a charge. Cannot be used with the ZoneModeling capability.
TimeModeling	Enables you to use time models when configuring a usage charge. The following subcapabilities specify which attributes of a time model can be used: • **Holiday:** Special day calendars • **DayOfWeek:** Days of the week • **TimeOfDay:** Time of day • **UsesEffectivity:** Effective periods
PriceSelector	(Mandatory if UsageBI or PriceOverrideBI is not included) Enables you to use price selectors based on event, service, or account attributes when configuring a charge. **AttributeBased:** This subcapability provides support for results based on specified event, service, or account attributes or a combination of such attribute types. Cannot be used with the UsageBI or PriceOverride capabilities.
UsageBI	(Mandatory if PriceSelector or PriceOverrideBI is not included) Enables you to configure one or more balance impacts for a usage charge. **MultiRum:** This subcapability enables you to select one or more RUMs in the **Measured By** field in the Create Charge dialog box. **StepsBasedOnUsageQuantity:** This subcapability enables you to create price steps based on event quantity. **MinimumCharge:** This subcapability enables you to set a minimum charge for each balance impact. **ChargingIncrement:** This subcapability enables you to specify a charging increment. **CreditWithoutValidity:** This subcapability enables you to specify a credit balance impact without using validity periods. **PriceTierUsesEffectivity:** This subcapability enables you to set effective dates for the prices in a charge. Cannot be used with the PriceSelector or PriceOverrideBI capabilities.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 36-4 (Cont.) Charging Capabilities in the BRM Offline Rating Target Engine Profile

Capability	Description
PriceOverrideBI	(Mandatory if PriceSelector or UsageBI is not included) Enables you to process prerated events by adding to or replacing the existing charge. Cannot be used with the PriceSelector and UsageBI capabilities.

Table 36-5 lists the discounting capabilities contained in this target engine profile:

Table 36-5 Discounting Capabilities in the BRM Offline Rating Target Engine Profile

Capability	Description
DiscountRateplanSelector	Enables you to use a discount selector when configuring a discount.
DiscountRateplan	(Mandatory) Enables you to configure discounts for a discount offer.
AbsoluteDateRange	(Mandatory) Enables you to specify fixed date ranges when configuring a discount.
DiscountOrChargeShareTrigger	Enables you to specify conditions that must be met before a discount can be applied to a charge.
DiscountOrChargeShareFilter	Enables you to specify criteria that a charge must satisfy to be eligible for a discount. This capability is included in the Standard Discount pricing profile but not in the Billing-time Discount pricing profile. **EventBasedChargeSelectorSpec:** This subcapability supports the use of event attributes to determine whether a charge qualifies for a particular discount.
DiscountOrChargeShareBI	(Mandatory) Enables you to configure one or more balance impacts for a discount.

Table 36-6 lists the charge sharing capabilities contained in this target engine profile:

Table 36-6 Charge Sharing Capabilities in the BRM Offline Rating Target Engine Profile

Capability	Description
ChargeShareRateplan	(Mandatory) Enables you to configure chargeshares for a chargeshare offer.
AbsoluteDateRange	(Mandatory) Enables you to specify fixed date ranges when configuring a chargeshare.
DiscountOrChargeShareTrigger	Enables you to specify conditions that must be met before a chargeshare can be applied to a charge.
DiscountOrChargeShareFilter	Enables you to specify criteria that a charge must satisfy to be eligible for a chargeshare. **EventBasedChargeSelectorSpec:** This subcapability supports the use of event attributes to determine whether a charge qualifies for a particular chargeshare.
DiscountOrChargeShareBI	(Mandatory) Enables you to configure one or more balance impacts for a chargeshare.

Subscription Engine Profile

The name of the subscription engine profile is RRE_SUBSCRIPTION.

This profile contains capabilities for the PRODUCT_OFFERING pricing type (see Table 36-7).

Copyright © 2017, 2026, Oracle and/or its affiliates.

See "About Pricing Types" for more information.

Table 36-7 lists the product offering capabilities contained in this engine profile:

Table 36-7 Product Offering Capabilities in the Subscription Engine Profile

Capability	Description
ChargeOffering	Enables you to configure a charge offer.
AlterationOffering	Enables you to configure a discount offer.
DistributionOffering	Enables you to configure a chargeshare offer.

ECE Rating Target Engine Profile

The name of the profile supported by ECE is ECE_RATING.

This profile contains capabilities for the following pricing types:

- USAGE_CHARGE (see Table 36-8)
- DISCOUNT (see Table 36-9)
- CHARGE_SHARE (see Table 36-10)

See "About Pricing Types" for more information.

Table 36-8 lists the charging capabilities contained in this target engine profile:

Table 36-8 Charging Capabilities in the ECE Rating Target Engine Profile

Capability	Description
ChargeRateplan	Enables you to configure charges for a charge offer.
SubscriberCurrency	(Mandatory) Enables you to specify the currency to use in the charge. **SingleCurrency:** This subcapability enables you to select only one currency in the **Currency** field in the Create Charge dialog box.
MultiRumGraph	Enables you to select one or more RUMs in the **Measured By** field in the Create Charge dialog box.
AbsoluteDateRange	Enables you to specify only fixed (not relative) date ranges when configuring a charge.
ZoneModeling	Enables you to use zone models when configuring a charge.
GenericSelector	Enables you to use generic selectors when configuring a charge.
TimeModeling	Enables you to use time models when configuring a usage charge. The following subcapabilities specify which attributes of a time model can be used: • **Holiday:** Special day calendars • **MonthOfYear:** Months of the year • **DayOfWeek:** Days of the week • **TimeOfDay:** Time of day • **DayOfMonth:** Days of month

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 36-8 (Cont.) Charging Capabilities in the ECE Rating Target Engine Profile

Capability	Description
UsageBI	(Mandatory) Enables you to configure one or more balance impacts for a usage charge. **SingleRum:** This subcapability enables you to select only one RUM in the **Measured By** field in the Create Charge dialog box. **StepsBasedOnBalanceElement:** This subcapability enables you to create price steps based on balance element quantity. **StepsBasedOnUsageQuantity:** This subcapability enables you to create price steps based on event quantity. **StepsBasedOnBalanceElementWithThreshold:** This subcapability enables you to create price steps based on balance element with threshold. **StepsBasedOnUsageQuantityWithThreshold:** This subcapability enables you to create price steps based on event quantity with threshold.

Table 36-9 lists the discounting capabilities contained in this target engine profile:

Table 36-9 Discounting Capabilities in the ECE Rating Target Engine Profile

Capability	Description
DiscountRateplan	(Mandatory) Enables you to configure discounts for a discount offer.
AbsoluteDateRange	(Mandatory) Enables you to specify fixed date ranges when configuring a discount.
DiscountOrChargeShareTrigger	Enables you to specify conditions that must be met before a discount can be applied to a charge.
DiscountOrChargeShareFilter	Enables you to specify criteria that a charge must satisfy to be eligible for a discount. This capability is included in the Standard Discount pricing profile but not in the Billing-time Discount pricing profile. **EventBasedChargeSelectorSpec:** This subcapability supports the use of event attributes to determine whether a charge qualifies for a particular discount.
DiscountOrChargeShareBI	(Mandatory) Enables you to configure one or more balance impacts for a discount.

Table 36-10 lists the charge sharing capabilities contained in this target engine profile:

Table 36-10 Charge Sharing Capabilities in the ECE Target Engine Profile

Capability	Description
ChargeShareRateplan	(Mandatory) Enables you to configure chargeshares for a chargeshare offer.
AbsoluteDateRange	(Mandatory) Enables you to specify fixed date ranges when configuring a chargeshare.
DiscountOrChargeShareTrigger	Enables you to specify conditions that must be met before a chargeshare can be applied to a charge.
DiscountOrChargeShareFilter	Enables you to specify criteria that a charge must satisfy to be eligible for a chargeshare. **EventBasedChargeSelectorSpec:** This subcapability supports the use of event attributes to determine whether a charge qualifies for a particular chargeshare.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 36-10 (Cont.) Charge Sharing Capabilities in the ECE Target Engine Profile

Capability	Description
DiscountOrChargeShareBI	(Mandatory) Enables you to configure one or more balance impacts for a chargeshare.

ECE Subscription Engine Profile

The name of the ECE subscription engine profile is ECE_SUBSCRIPTION.

This profile contains capabilities for the PRODUCT_OFFERING pricing type (see Table 36-11).

See "About Pricing Types" for more information.

Table 36-11 lists the product offering capabilities in this engine profile:

Table 36-11 Product Offering Capabilities in the ECE Subscription Engine Profile

Capability	Description
ChargeOffering	Enables you to configure a charge offer.
DiscountOffering	Enables you to configure a discount offer.
ChargeShareOffering	Enables you to configure a chargeshare offer.

Target Engine Profiles for the Default Pricing Profiles

The pricing capabilities (features) in a pricing profile are based on the capabilities in the target engine profiles with which it is associated. A pricing profile can be associated with one or more target engine profiles (see "About Associating Custom Pricing Profiles with Target Engine Profiles").

Table 36-12 shows the default target engine profile or profiles for each default pricing profile:

Table 36-12 Target Engine Profiles Associated with the Default Pricing Profiles

Default Pricing Profile	Default Target Engine Profile	Target Engine Profile Pricing Capabilities
Online Usage	RRE_RATING	See Table 36-3.
Offline Usage	BRE_RATING	See Table 36-4.
Common Usage	RRE_RATING and BRE_RATING	See Table 36-3 and Table 36-4.
Standard Discount	BRE_RATING	See Table 36-5.
Billing-time Discount	BRE_RATING	See Table 36-5.
ChargeShare	BRE_RATING	See Table 36-6.
Convergent Usage	ECE_RATING	See Table 36-8.
Standard Discount	ECE_RATING	See Table 36-9.
Standard Chargesharing	ECE_RATING	See Table 36-10.

Copyright © 2017, 2026, Oracle and/or its affiliates.

About Associating Custom Pricing Profiles with Target Engine Profiles

Before you can publish pricing components that use a custom pricing profile to a rating engine, you must associate the pricing profile with at least one target engine profile. The associated target engine profiles determine which rating or subscription engines the components are published to.

The target engines must support all the capabilities in the pricing profile. See the tables in "About the PDC Target Engine Profiles" for the capabilities supported by each PDC target engine profile.

> **Note**
>
> When a component that uses a pricing profile is submitted for publication, PDC validates the pricing profile against its associated target engine profiles to ensure that the capabilities in the pricing profile are supported by the target engines.

If you associate a pricing profile with multiple target engine profiles, the pricing profile must contain only capabilities that are supported by *all* of the associated target engine profiles. See "About Pricing Profiles" for more information.

If a pricing profile is not associated with a target engine profile, any pricing components that use that profile are not published to a target engine.

See "Target Engine Profiles for the Default Pricing Profiles" for information about the target engine profiles associated with the default pricing profiles.

See "Associating Custom Pricing Profiles with Target Engine Profiles" for instructions on how to associate pricing profiles with target engine profiles.

About Importing Custom Pricing Profiles into PDC

You define a custom pricing profile in an XML file and use the **ImportExportPricing** utility to import the XML file into PDC. The XML file containing the custom pricing profile must conform to the format specified in the **profiles.xsd** file in *PDC_home***/apps/xsd**.

See "Prerequisites for Using the ImportExportPricing Utility" for information about prerequisites for using the **ImportExportPricing** utility.

Setting Up Custom Pricing Profiles

To set up a custom pricing profile in PDC:

1. Create a custom pricing profile (see "Creating Custom Pricing Profiles").
2. Associate the profile with a target engine profile (see "Associating Custom Pricing Profiles with Target Engine Profiles").
3. Import the profile into PDC (see "Importing Custom Pricing Profiles into PDC").

Creating Custom Pricing Profiles

To create a custom pricing profile:

Copyright © 2017, 2026, Oracle and/or its affiliates.

1. Make a copy of the default pricing profile to customize.

   See "About Default Pricing Profiles" for descriptions and locations of the default pricing profiles.

2. Open the copy in a text or XML editor.

3. In the copy, modify the pricing profile name to distinguish the copy from the original:

   ```
   <pricingProfile>
      <name>CustomPricingProfileName</name>
      . . .
   ```

   where *CustomPricingProfileName* is the new name for the copy.

4. Add or delete pricing capabilities, as necessary.

   Note the following:

   - If only one subcapability exists for a particular capability in the target engine profile, the pricing profile must include that subcapability.
   - If multiple subcapabilities exist for a particular capability in a target engine profile, at least one of the subcapabilities must be included in the pricing profile. The custom pricing profile does not, however, have to use the same subcapability as the default pricing profile.

   See "About Customizing Pricing Profiles" for more information.

5. Save your changes.

Associating Custom Pricing Profiles with Target Engine Profiles

To associate a custom pricing profile with a target engine profile:

1. Open the file containing the custom pricing profile in a text or XML editor.

2. Add the following entry to the **<profiles>** element:

   ```
   <profileUsedBy>
      <pricingProfileName>CustomPricingProfile</pricingProfileName>
      <targetEngineProfileName>TargetEngineProfile1</targetEngineProfileName>
      <targetEngineProfileName>TargetEngineProfile2</targetEngineProfileName>
      <priceListName>Default</priceListName>

   </profileUsedBy>
   ```

   where:

   - *CustomPricingProfile* is the name of the pricing profile that you are associating with the target engine profile.
   - *TargetEngineProfile#* is the name of the target engine profile that you are associating with the custom pricing profile.

Copyright © 2017, 2026, Oracle and/or its affiliates.

> **Note**
>
> For each additional target engine that you want to associate with the pricing profile, add another **<targetEngineProfileName>** entry to the **<profileUsedBy>** element.

3. Save and close the file.

See "About Associating Custom Pricing Profiles with Target Engine Profiles" for more information.

Importing Custom Pricing Profiles into PDC

To import a custom pricing profile into PDC, you must have the **Pricing Design Admin** role.

To import a custom pricing profile into PDC:

1. Verify that the custom pricing profile XML file is complete and follows the guidelines specified in the **profiles.xsd** file in the *PDC_home***/apps/xsd** directory.
2. Go to the *PDC_home***/apps/Samples/Examples** directory, and enter the following command:

   Linux

   `ImportExportPricing -import -profile` *`ProfileFileName`*

   where *ProfileFileName* specifies the full path and file name of the XML file containing the custom profile data.

> **Note**
>
> To run this command, you must have *PDC_home***/apps/bin** in your PATH environment variable.

For example, the following command imports a custom pricing profile from the **MyPricingProfile.xml** file into the PDC database in a Linux environment:

```
ImportExportPricing -import -profile /MyPricingProfile.xml
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

37

Configuring Product Specification Attributes for Pricing Components

Learn how to configure product specification attributes for pricing components created in XML, or in Pricing Design Center for Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

- About Product Specification Attributes for Pricing Components
- Configuring Product Specification Attributes for Pricing Components
- Using Product Specification Attributes in Pricing Components
- About Validating Product Specification Attributes for Pricing Components
- Modifying Product Specification Attribute Definitions

About Product Specification Attributes for Pricing Components

You can use product specification attributes on pricing components to save extra information that may be useful to external applications.

In the TM Forum data model, a product specification is a collection of attributes or product specification characteristics that describe a product. Although BRM and PDC do not use TM Forum product specifications or act on product specification attributes, you can choose to store the attribute information in the BRM database so that it can be queried and used by external applications.

In BRM, you create product specification attributes in a configuration template file and then provide values for them while creating your pricing components in PDC or in XML.

For example, if your external enterprise product catalog has an attribute for tracking the lifecycle status of a charge offer with relevant values beyond the two statuses offered by PDC (draft or promoted), you can use a product specification attribute to store that status. Your enterprise product catalog can then query or update that status as needed.

At a high level, the steps for using product specification attributes with pricing components are as follows:

1. Create product specification attributes in a configuration template file. See "Configuring Product Specification Attributes for Pricing Components".
2. Import the template file by using the **ImportExportPricing** utility. See "Importing Pricing and Setup Components".
3. Create pricing components in PDC or XML, including the product specification attribute values. See "Using Product Specification Attributes in Pricing Components".
   When you create pricing components in PDC, the UI validates the values against the configuration template file.

   When you import pricing components from an XML file using **ImportExportPricing**, the utility validates the values in the pricing component files against the configuration template

Copyright © 2017, 2026, Oracle and/or its affiliates.

file. The import fails if any values don't conform to the configuration file. See "About Validating Product Specification Attributes for Pricing Components".

Afterward, if you need to change your product specification attributes, follow the instructions in "Modifying Product Specification Attribute Definitions".

Configuring Product Specification Attributes for Pricing Components

You configure product specification attributes for pricing components in a **ProductSpecCharacteristicTemplate.xml** configuration file and import them along with other pricing setup components using the **ImportExportPricing** utility.

PDC provides a sample XML file for product specification attributes in the *PDC_home***/apps/Samples/Examples** directory, where *PDC_home* is the directory in which you installed PDC.

In the configuration file, you use a separate template entry for each pricing component type. You can create template entries for the pricing components listed in Table 37-1.

Table 37-1 Pricing Components That Support Product Specification Attributes

Pricing Component	pricingObjectType Value
Charge offer	CHARGE_OFFERING
Discount offer	ALTERATION_OFFERING
Chargeshare offer	DISTRIBUTION_OFFERING
Bundle	BUNDLED_PRODUCT_OFFERING
Package	PACKAGE_OBJ

You define the following elements for each attribute:

- **name**: The name of the attribute as it will appear when defining the pricing component. The name can be in any language using UTF-8 format.
- **type**: The type of value for the attribute. Can be:
  - ANY
  - BOOLEAN
  - DATE
  - NUMBER
- **optional**: Whether the attribute is optional (**true**) or not (**false**).
- **values**: A configured list of possible values for the attribute. If you provided values here, the import of pricing components will fail if an attribute contains a value that was not defined here. If you don't provide values here, the pricing components can include any value that matches the type. The values can be in any language using UTF-8 format.
- **description**: A description of the attribute. The description can be in any language using UTF-8 format.
- **cardinality**: Whether only one value can be specified for the attribute when creating the pricing component (**SINGLE**), or multiple values can be specified (**MULTIPLE**).

Copyright © 2017, 2026, Oracle and/or its affiliates.

The following example shows the XML file to create two lifecycle status attributes for a charge offer:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<cim:ConfigObjects xmlns:cim="http://xmlns.oracle.com/communications/platform/
model/Config">
    <productSpecCharacteristicTemplates xmlns:cim="http://xmlns.oracle.com/
communications/platform/model/Config">
        <name>Charge Offer Product Specification Attribute Template</name>
        <internalId>1234</internalId>
        <priceListName>Default</priceListName>
        <obsolete>false</obsolete>
        <pricingObjectType>CHARGE_OFFERING</pricingObjectType>
        <productSpecCharacteristics>
            <name>Status</name>
            <type>ANY</type>
            <optional>true</optional>
            <values>Draft</values>
            <values>Test</values>
            <values>Release</values>
            <values>Suspend</values>
            <values>Retired</values>
            <description>Extended lifecycle status for the charge offer.</
description>
            <cardinality>SINGLE</cardinality
        </productSpecCharacteristics>
        <productSpecCharacteristics>
            <name>Allowed Locations</name>
            <type>ANY</type>
            <optional>true</optional>
            <values>ASIA</values>
            <values>USA</values>
            <values>EUROPE</values>
            <values>CANADA</values>
            <values>OCEANIA</values>
            <description>Locations where the charge offer can be sold.</
description>
            <cardinality>MULTIPLE</cardinality
        </productSpecCharacteristics>
    </productSpecCharacteristicTemplates>
</cim:ConfigObjects>
```

Using Product Specification Attributes in Pricing Components

When creating pricing components in PDC or XML, you can use product specification attributes configured in the **ProductSpecCharacteristicTemplate.xml** file.

In PDC, any product specification attributes you configure in the template file appear in the Product Spec Characteristic Attributes (Custom Attributes) section for charge offers, discount offers, chargeshare offers, bundles, and packages. You can add or update the values.

Copyright © 2017, 2026, Oracle and/or its affiliates.

In the XML file for your pricing component, use the **productSpecCharacteristic** element to add an attribute, and provide the **name** and a **value** element for each attribute. For example:

```
<chargeOffering externalID="1234">
      <name>ChargeOfferWithExtendedAttribute</name>
      <internalId>5678</internalId>
      <pricingProfileName>Product Offering</pricingProfileName>
      <priceListName>Default</priceListName>
      <obsolete>false</obsolete>
      <timeRange>0/inf</timeRange>
      <productSpecName>TelcoGsmTelephony</productSpecName>
      <productSpecCharacteristic>
         <name>Status</name>
         <value>Test</value>
      </productSpecCharacteristic>
      <productSpecCharacteristic>
         <name>Allowed Locations</name>
         <value>ASIA</value>
         <value>USA</value>
         <value>EUROPE</value>
      </productSpecCharacteristic>
...
```

Keep the following in mind when defining values for product specification attributes:

- You can only use the attributes configured for the template with the same pricing type as the pricing component you are creating. In the example above, because you are creating a charge offer, all of the product specification attributes must come from the template configured for the CHARGE_OFFERING pricing type.
- Values are case sensitive.
- The value in the **name** must match one of the attributes you configured in the template file.
- The valid values are determined by the **type** and **value** elements configured for the attribute in the template file. Table 37-2 describes the supported types.

Table 37-2 Supported Value Types

Configured Type	Supported Values
ANY	UTF-8 strings in any language. If any **value** elements were configured for the attribute in the template file, only those values are supported.
DATE	Dates in format YYYYMMDD. If any **value** elements were configured for the attribute in the template file, only those values are supported.
BOOLEAN	**true** or **false**
NUMBER	Numbers 0-9. Decimals are supported, using either a period or comma as a decimal separator, according to your localization settings. If any **value** elements were configured for the attribute in the template file, only those values are supported.

See "About Validating Product Specification Attributes for Pricing Components".

Copyright © 2017, 2026, Oracle and/or its affiliates.

> ⚠ **Caution**
>
> After importing your pricing components, if you update them using **ImportExportPricing**, you must include all product specification attributes names and values in the definition, regardless of whether you are updating them. Any product specification attributes you do not include when you import the updated file will be removed from the pricing component.

About Validating Product Specification Attributes for Pricing Components

The validation method for product specification attributes depends on where you create pricing components.

When you create pricing components in PDC, the UI validates the values of product specification attributes against the attribute configuration template. Error messages are displayed for any failed validations.

When you import pricing components using the **ImportExportPricing** utility, the utility validates that:

- A template exists in the configuration file for the pricing component type.
- The name of an attribute appears in the template for the pricing component. For example, an attribute specified for a discount offer must appear under the template for the ALTERATION_OFFERING price type.
- Each attribute appears only once in the pricing component definition.
- The value of the attribute matches the **type** element configured in the template. For example, an attribute with the BOOLEAN type can only have **true** or **false** as values.
- The value of the attribute matches one of the **value** elements configured in the template if the attribute configuration contains a specified list of values.
- A value is provided for any attributes where the **optional** element is set to **false**.

The UI and the utility only validate against the configuration file. They do not validate based on the meaning of the attributes. For example, the UI or utility cannot validate that the start date is earlier than the end date in product specification attributes for start and end dates, or that a location in a product specification attribute for allowed locations matches the currency defined for a pricing component.

You can optionally implement custom validations for charge offers, discount offers, and bundles created in XML using the following pricing policy opcodes:

- PCM_OP_PRICE_POL_VALID_PRODUCT
- PCM_OP_PRICE_POL_VALID_DISCOUNT
- PCM_OP_PRICE_POL_VALID_DEAL

See "About the PREP and VALID Opcodes" in *BRM Developer's Guide* for information about using VALID opcodes.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Modifying Product Specification Attribute Definitions

If you want to modify any existing product specification attribute definitions, you must do the following:

- Export, update, and then re-import the attribute configuration template.
- Export, update, and then re-import your pricing components, which have been modified to comply with the updated attribute configuration template.

Following this procedure ensures that any changes made to your attribute definitions do not invalidate the pricing components already stored in PDC.

The following example shows how to modify the attribute definitions for charge offers. You can use a similar procedure to modify attribute definitions for discount offers and charge share offers.

1. Export all charge offers from PDC. For example:

   ```
   ImportExportpricing -export MyChargeOffers -config CHARGE_OFFERING
   ```

   In this example, all charge offers are exported to the **MyChargeOffers_export_pricing.xml** file.
2. Export all attribute configuration template objects from PDC. For example:

   ```
   ImportExportPricing -export ProductSpecCharacteristicTemplate -config
   PSCTEMPLATE
   ```

   In this example, all attribute configuration template objects are exported to the **ProductSpecCharacteristicTemplate_export_config.xml** file.
3. Update your attribute configuration template XML file according to the information in "Configuring Product Specification Attributes for Pricing Components".
4. Import your modified attribute configuration template XML file into PDC. For example:

   ```
   ImportExportPricing -import -config
   ProductSpecCharacteristicTemplate_export_config.xml -ow
   ```

5. Update the charge offers in your **MyChargeOffers_export_pricing.xml** file to comply with the latest attribute configuration template.
6. Import your updated charge offers into PDC. For example:

   ```
   ImportExportPricing -import -pricing MyOffers_export_pricing.xml -ow
   ```


Copyright © 2017, 2026, Oracle and/or its affiliates.

38

Replicating Data Between PDC Systems

Learn how to replicate Pricing Design Center (PDC) data from one Oracle Communications Billing and Revenue Management (BRM) system to another, such as from a development system to a test system.

Topics in this document:

- About Replicating Data
- Replicating Setup Components Managed in BRM
- Replicating Data Between PDC Databases

About Replicating Data

You replicate PDC data by extracting data from a *source* PDC database and then loading the data into a *destination* PDC database by using the **ImportExportPricing** utility. See "Importing and Exporting Pricing and Setup Components" for more information.

The target system must include both the BRM server components and a PDC database. If you create a new BRM installation, do not load the sample files in the *PDC_home***/apps/samples/examples** directory after installing PDC, where *PDC_home* is the directory in which the PDC software is installed.

Replicating PDC data between two BRM systems involves the following tasks:

1. Replicating Setup Components Managed in BRM
2. Replicating Data Between PDC Databases

Replicating Setup Components Managed in BRM

Some setup components are managed in BRM and are then synchronized with PDC.

To ensure that all the setup components managed in the *source* BRM system are synchronized with the *destination* BRM and PDC systems, extract the setup components from the *source* BRM database and load them into the *destination* BRM database by using the BRM load utilities. When you replicate the PDC data, these components will be synchronized with the *destination* PDC system.

Table 38-1 lists the type of setup components that are managed in BRM and their corresponding load utilities.

Table 38-1 Setup Components Managed in BRM and Their Corresponding Load Utilities in BRM

Setup Component Type	BRM Load Utility
Service	**pin_deploy**
Account	**pin_deploy**
Event	**pin_deploy**

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 38-1 (Cont.) Setup Components Managed in BRM and Their Corresponding Load Utilities in BRM

Setup Component Type	BRM Load Utility
G/L ID	**load_pin_glid**
Tax Supplier	**load_tax_supplier**
Provisioning Tag	**load_config_provisioning_tags**

Replicating Data Between PDC Databases

To replicate data:

1. Extract the service, account, event, and profile attribute specifications from the *source* PDC database and load them into the *destination* PDC database by doing the following:
   a. Export the service, account, event, and profile (if applicable) attribute specifications from the *source* PDC database by running the following command:

      ```
      ImportExportPricing -export -metadata
      ```

      The service, account, event, and profile (if applicable) attribute specifications are exported to the **export_meta_config.xml** file.

      > **Note**
      >
      > Do not change the internal IDs of the exported attribute specifications.

   b. Ensure that the transformation engines are running. See the discussion about starting the transformation engines in *PDC Installation Guide* for more information.
   c. Import the exported attribute specifications from the **export_meta_config.xml** file into the *destination* PDC database by running the following command:

      ```
      ImportExportPricing -import -metadata export_meta_config.xml
      ```

2. Run the **SyncPDC** utility by doing the following:
   a. Go to the *BRM_Integration_Pack_Home***/apps/syncpdc** directory.
   b. Enter the following command:

      ```
      startSyncPDC
      ```

      The **Enter Key Password** prompt appears.
   c. Enter the password PDC uses for accessing the PDC alias key in the keystore (*BRM_Integration_Pack_Home***/apps/conf/pdc.jks**).
3. Extract all the setup components that are defined in PDC from the *source* PDC database and load them into the *destination* PDC database by doing the following:
   a. Export all the setup components from the *source* PDC database by running the following command:

      ```
      ImportExportPricing -export -config
      ```

      The setup components are exported to the **export_config.xml** file.

Copyright © 2017, 2026, Oracle and/or its affiliates.

b. Import the exported setup components from the **export_config.xml** file into the *destination* PDC database by running the following command:

```
ImportExportPricing -import -config export_config.xml
```

4. Extract all the pricing components from the *source* PDC database and load them into the *destination* PDC database by doing the following:

   a. Export all the pricing components from the *source* PDC database by running the following command:

   ```
   ImportExportPricing -export -pricing
   ```

   The pricing components are exported to the **export_pricing.xml** file.

   b. Import all the pricing components from the **export_pricing.xml** file into the *destination* PDC database by running the following command:

   ```
   ImportExportPricing -import -pricing export_pricing.xml
   ```


Copyright © 2017, 2026, Oracle and/or its affiliates.

Part VII
Pricing Utilities

This part describes the utilities provided with Oracle Communications Billing and Revenue Management (BRM) Pricing Design Center (PDC).

This part contains the following chapters:

- Pricing Utilities

Copyright © 2017, 2026, Oracle and/or its affiliates.

39

Pricing Utilities

Learn about the syntax and parameters for the Oracle Communications Pricing Design Center (PDC) utilities.

Topics in this document:

- ImportExportPricing
- PDCWalletUtil.sh
- PDCBRMWalletUtil.sh
- load_pin_snowball_distribution
- load_pin_spec_rates
- load_pin_sub_bal_contributors
- pin_discount_cleanup

ImportExportPricing

Use the **ImportExportPricing** utility to import and export pricing and setup components from the PDC database by using XML files. See "Importing and Exporting Pricing and Setup Components" for information on using this utility.

The XML files that are used for importing or exporting data must conform to the format detailed in the XML schema definition (XSD) files.

The utility uses separate XSD files for importing or exporting pricing components, setup components, and profile data. The XSD files are available in their respective directories in the *PDC_home***/apps/xsd** directory, where *PDC_home* is the directory in which the PDC software is installed. See "About the XSD Files" for more information.

Location

*PDC_home***/apps/bin**

Syntax Overview

You can run the **ImportExportPricing** utility in the following modes:

- **export**: Export pricing and setup components from the PDC database into XML files. See "Export Mode".
- **import**: Import pricing and setup components data from an XML file into the PDC database. See "Import Mode".
- **changeSetStatus**: Check the status of one or more changesets. See "Changeset Status Mode".
- **publish**: Publish setup components from PDC to the specified target engines, such as batch rating engine, real-time rating engine, and Oracle Communications Elastic Charging Engine (ECE). See "Publish Mode".

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **keep**: Retain the latest versions of successfully promoted PDC components, but delete old versions. See "Keep Mode".
- **delete**: Delete pricing profiles or pricing components that are managed in Oracle Communications Billing and Revenue Management (BRM). See "Delete Mode".
- **type**: Display the list of setup or pricing component types supported in PDC. See "Type Mode".

Results

The **ImportExportPricing** utility notifies you when it runs successfully. Otherwise, look in the **ImportExportPricing**_*username_timestamp*.**log** file, where *username* is the name of user who used the utility and *timestamp* is the timestamp of the log file in the format *yyyy*-*mm*-*dd_hh*-*mm*-*ss*. This file is in the directory from which the utility was started.

Export Mode

The **export** mode exports pricing and setup components from the PDC database into XML files.

export Mode Syntax

```
ImportExportPricing -export [Prefix] -config [SetupObjectType1,SetupObjectType2]
                                     -metadata [MetaDataObjectType1,MetaDataObjectType2]
                                     -pricing [PricingObjectType1,PricingObjectType2]
                                     -brmobject [brmObjectType1,brmObjectType2]
                                     -profile
                                     -customfields
                                     [-separator "SeparatorCharacter"]
                                     [-n "ObjectName1,ObjectName2"]
                                     [-pricingprofilename "Name1,Name2"]
                                     [-productSpecName ProductSpecName1,ProductSpecName2]
                                     [-expRefs|-expAllRefs] [-inclFailedObjs]
                                     [-expObsolete]
                                     [-ma|-modifiedAfter time]
                                     [-mby|modifiedBy user1,user2]
                                     [-l|-LogLevel level] [-v|-verbose] [-ow|-overwrite]
                                     -appsvruser ApplicationServerUserName
                                     -pdcuser PdcUserName -url appSvrUrl
                                     [-h|-help]
```

export Mode Parameters

-export [*Prefix*]
Exports the data from a PDC database and generates a separate XML file for pricing components, setup components, and pricing profile data in the directory from which you run the utility. The utility generates the file names as:

- *Prefix*_**export_pricing.xml** for the file containing pricing components.
- *Prefix*_**export_config.xml** for the file containing setup components.
- *Prefix*_**export_profile.xml** for the file containing pricing profiles data.

If you do not specify *Prefix*, the utility generates the file name as follows:

- **export_pricing.xml** for the file containing pricing components. If this file already exists in PDC, the utility generates the file name as **export_pricing**_*timestamp*.**xml**, where *timestamp* is the server's local time in the format *yyyy*-*mm*-*dd_hh*-*mm*-*ss*.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **export_config.xml** for the file containing setup components. If this file already exists in PDC, the utility generates the file name as **export_config**_*timestamp*.**xml**, where *timestamp* is the server's local time in the format *yyyy-mm-dd_hh-mm-ss*.
- **export_profile.xml** for the file containing pricing profile data. If this file already exists in PDC, the utility generates the file name as **export_profile**_*timestamp*.**xml**, where *timestamp* is the server's local time in the format *yyyy-mm-dd_hh-mm-ss*.

-config [*SetupObjectType1*, *SetupObjectType2*,…]
Exports all setup components that are defined in PDC from the PDC database to the XML file.
Exports *SetupObjectType* from the PDC database to the XML file, where *SetupObjectType* is the setup component type listed in Table 39-1.
If you do not specify *SetupObjectType*, the utility exports all setup components from the PDC database into the XML file.

Table 39-1 Setup Component Types and the Setup Components They Represent

Setup Component Type	Setup Component
VALUE_MAP	Value Map
EVENT_ATTRIBUTE_SPEC	Event
PRODUCT_ATTRIBUTE_SPEC	Service
CUSTOMER_ATTRIBUTE_SPEC	Account Attribute
RUM_CONFIGURATION	RUM
ZONE_RESULT_CONFIGURATION	Impact Category for Zones
ATTRIBUTE_SPEC_MAP	Service-Event Map
BALANCE_ELEMENT	Balance Element
GLID	G/L ID
TAX_CODE	Tax Code
TAX_EXEMPTION_CODE	Tax Exemption Code
TAX_SUPPLIER	Tax Supplier
GEO_ZONE_MODEL	Geographical Zone Model
STANDARD_ZONE_MODEL	Standard Zone Model
PROVISIONING_TAG	Provisioning Tag
HOLIDAY_CALENDAR	Special Day Calendar
CUSTOM_ANALYZER_RULE	Custom Rule
PROFILE_ATTRIBUTE_SPEC	Profile
DELIVERABLE	Deliverable
PSCTEMPLATE	Extended Attribute Template
PRICETAGCONFIG	Price Tag Configuration

-metadata [*MetaDataObjectType1*, *MetaDataType2*,…]
Exports the metadata object types specified in Table 39-2 from the PDC database into the XML file.
If you do not specify *MetaDataObjectType*, the utility exports all metadata components from the PDC database into the XML file.

Table 39-2 Metadata Component Types and the Metadata Components They Represent

Metadata Component Type	Metadata Component
EVENT_ATTRIBUTE_SPEC	Event
PRODUCT_ATTRIBUTE_SPEC	Service

Copyright © 2017, 2026, Oracle and/or its affiliates.

Table 39-2 (Cont.) Metadata Component Types and the Metadata Components They Represent

Metadata Component Type	Metadata Component
PROFILE_ATTRIBUTE_SPEC	Profile attribute
CUSTOMER_ATTRIBUTE_SPEC	Account attribute

-pricing [*PricingObjectType1*, *PricingObjectType2*,…]
Exports *PricingObjectType* from the PDC database to the XML file, where *PricingObjectType* is the pricing component type listed in Table 39-3.
If you do not specify *PricingObjectType*, the utility exports all pricing components from the PDC database into the XML file.

Table 39-3 Pricing Component Types and the Pricing Components They Represent

Pricing Component Type	Pricing Component Represented
CHARGE_RATE_PLAN	Charge
ALTERATION_RATE_PLAN	Discount
DISTRIBUTION_RATE_PLAN	Chargeshare
ROLLOVER_RATE_PLAN	Rollover
ALTERATION_RATE_PLAN_SELECTOR	Discount Selector
GENERIC SELECTOR	Generic Selector
CHARGE_RATE_PLAN_SELECTOR	Charge Selector
PRICE_MODEL_SELECTOR	Price Selector
APN_MAP	APN Selector
USC_MAP	USC Selector
TAX_SELECTOR	Tax Selector
TAX_EXEMPTION_SELECTOR	Tax Exemption Selector
TIME_MODEL	Time Model
TRIGGER_SPEC	Trigger
CHARGE_SELECTOR_SPEC	Filter
ALTERATION_POP_MODEL	Discount Pricing
ROLLOVER_POP_MODEL	Rollover Rules
RECURRING_POP_MODEL	Recurring Pricing
ONE_TIME_POP_MODEL	One-Time Pricing
CHARGE_OFFERING	Charge Offer
ALTERATION_OFFERING	Discount Offer
DISTRIBUTION_OFFERING	Chargeshare Offer
ALTERATION_EXCLUSION	Discount Exclusion
SUBSCRIPTION_TERMS	Subscription Terms
SUBSCRIPTION_TERMS_AUTO_RENEW	Subscription Terms with Automatic Renewal
BUNDLED_PRODUCT_OFFERING	Bundle
BUNDLE_TRANSITION	Bundle Transition
PACKAGE_OBJ	Package
PACKAGE_EXCLUSION	Discount Restriction
PACKAGE_TRANSITION	Package Transition
PACKAGE_LIST	Package List

Copyright © 2017, 2026, Oracle and/or its affiliates.

-brmObject [*brmObjectType1*, *brmObjectType2*,...]
Exports BRM-mastered configuration objects from the PDC database into the XML file. This includes services, events, account attributes, tax suppliers, and G/L IDs.

-profile
Exports pricing profile data from the PDC database into the XML file.

-customfields
Exports custom field data from the PDC database into the XML file.

-separator "*SeparatorCharacter*"
Specifies the character used to split the names of pricing and configuration objects to be exported from the PDC database. This option must be provided before the **name** parameter in the **ImportExportPricing** command. If not specified, the default separator is a comma (**,**). Use this option when object names contain commas to ensure correct parsing.

-n "*ObjectName1*, *ObjectName2*,..."
Filters the pricing or setup components based on the specified component names. You can use an asterisk (*) at the end of the name to search for variations of that name. For example, searching for **chargeoffergsm*** yields variations of that name, such as **chargeoffergsm1** and **chargeoffergsm2**.

> **Note**
>
> When you use the **-n** parameter, you must specify either **-config** or **-pricing**.
> The parameters **-n** and **-productSpecName** are mutually exclusive.

-pricingprofilename "*Name1*, *Name2*,..."
Exports pricing components scoped to the pricing profile names specified in the list.
If this parameter is used with the **-pricing** parameter when a list of pricing component types is specified, the utility exports all the pricing components of the specified type scoped to the pricing profiles names specified in the list.
If this parameter is specified with the **-pricing** parameter when no pricing component types are specified, the utility exports all the pricing components that are scoped to a pricing profile.

-productSpecName *ProductSpecName1, ProductSpecName2,...*
Filters the exported pricing components that are scoped to the specified services.
If you specify this parameter along with **-pricing** *PricingObjectType*, the utility exports all pricing component types that are scoped to *ProductSpecName*. You can use this parameter only for the following pricing component types:

- ALTERATION_OFFERING
- CHARGE_OFFERING
- DISTRIBUTION_OFFERING
- CHARGE_RATE_PLAN

If you specify this parameter without specifying *PricingObjectType*, the utility exports all pricing components scoped to the specified service.

Copyright © 2017, 2026, Oracle and/or its affiliates.

> **Note**
>
> The parameters **-n** and **-productSpecName** are mutually exclusive.

-expRefs
Exports all pricing components referenced by *PricingObjectType*. For example, if multiple charge offers are referenced by bundles, this parameter exports all charge offers when exporting bundles. The utility ignores this parameter if it is used with the **-config** or **-metadata** parameter.

-expAllRefs
Exports all PDC components (including metadata, setup components, and pricing components) referenced by *PricingObjectType*. For example, if setup components, such as ratable usage metrics (RUMs) and tax codes are referenced by charge offers, then this parameter exports RUMs and tax codes when exporting charge offers. This parameter applies only to the **-pricing** parameter.

-inclFailedObjs
Exports the setup or pricing objects that were last promoted objects (including those with a failed changeset status). By default, the utility exports only successfully transformed objects. The utility ignores this parameter if it is used with the **-metadata** parameter.

-expObsolete
Exports obsolete pricing and setup data along with the other pricing and configuration data. By default, the utility does not export obsolete pricing and setup data.

-ma *time* | -modifiedAfter *time*
Exports pricing components modified on or after the specified *time*. Enter the *time* in the format *yyyy-mm-dd_hh-mm-ss*. For example: 2010-01-05T19:05:09GMT+05:30.

-mby *user1, user2,…* | -modifiedBy *user1, user2,...*
Exports the pricing components created or modified by the specified users.

> **Note**
>
> This parameter and **-productSpecName** are mutually exclusive.

-l *loglevel* | -loglevel *loglevel*
Specifies how much information the utility should log. The logs are stored in the **ImportExportPricing**_*username_timestamp*.**log** file, where *username* is the name of user who used the utility and *timestamp* is the time the log file was created in the format *yyyy-mm-dd_hh-mm-ss*.
Set *loglevel* to one of the following:

- FINE
- FINER
- FINEST
- INFO
- OFF
- SEVERE

Copyright © 2017, 2026, Oracle and/or its affiliates.

- WARNING

The default is **INFO**.

-v | -verbose
Displays information about successful or failed processing as the utility runs.

-ow | -overwrite
Overwrites any existing XML file containing the exported pricing or setup components with the file specified in the **-export [***Prefix***]** parameter.

-appsvruser *ApplicationServerUserName*
Specifies the user name to access the Oracle WebLogic Server. If you do not specify this parameter, the utility prompts you to enter the user name at the command prompt.

-pdcuser *PdcUserName*
Specifies the user name to access PDC. If you do not specify this parameter, the utility prompts you to enter the user name at the command prompt.

-url *appSvrUrl*
Specifies the application server URL, which has the format **t3://***hostname***:***port*.

-h
Displays the syntax for this utility.

Import Mode

The **import** mode imports the component data from or more XML files into the PDC database. See "Importing Pricing and Setup Components" for more information.

import Mode Syntax

```
ImportExportPricing -import  -metadata FileOrDirectoryName
                             -config FileOrDirectoryName
                             -pricing FileOrDirectoryName
                             -profile FileName
                             -customfields FileName
                             -ignoreID
                             -submitCS
                             [-l | -loglevel loglevel]
                             [-v | -verbose]
                             [-ow | -overwrite]
                             [-changeSetName ChangeSetName]
                             -appsvruser ApplicationServerUserName
                             -pdcuser PdcUserName -url appSvrUrl
                             [-h | -help]
```

import Mode Parameters

-import
Imports data from one or more XML files that conform to the format detailed in the XSD file to the PDC database. In addition to the **-import** parameter, you must also include the **-metadata**, **-config**, **-pricing**, **-profile**, or **-customfields** parameter.

-metadata *FileOrDirectoryName*
Imports the metadata objects from *FileOrDirectoryName* to the PDC database. *FileOrDirectoryName* specifies either the full path and file name of the import XML file or the full path to the directory in which your import XML files reside. Ensure the user has at least

Copyright © 2017, 2026, Oracle and/or its affiliates.

read permission for the input directory, its subdirectories, and its files. See "Importing Multiple XML Files from a Directory" for more information.

-config *FileOrDirectoryName*
Imports the setup components from *FileOrDirectoryName* to the PDC database.

-pricing *FileOrDirectoryName*
Imports the pricing components from *FileOrDirectoryName* to the PDC database.

-profile *FileName*
Imports the profile data from *FileName* to the PDC database, where *FileName* is the name and path to the XML file that contains your profile data.
If you specify **-pricing** and **-config** parameters along with **-profile**, the utility imports profile data and setup components before importing the pricing components. If the import of the profile data and setup components is not successful, the utility does not import the pricing components.

-customfields *FileName*
Imports the profile data from *FileName* to the PDC database.

-ignoreID
Ignores the internal IDs in the specified XML file. The utility looks up the internal IDs by object name or generates internal IDs for new objects. Use this option to avoid conflicts when importing data from another environment.
Use this parameter with the **-ow** parameter when importing existing data.

-submitCS
Creates new or modified pricing data and adds it to a changeset for the user to process at a later time. When this option is not specified, the changes to pricing data in the changeset are committed and published to the target engine.

-l *loglevel*
Specifies how much information the utility should log. The logs are stored in the **ImportExportPricing**_*username_timestamp*.**log** file, where *username* is the name of user who used the utility and *timestamp* is the time the log file was created in the format *yyyy-mm-dd_hh-mm-ss*.
Set *loglevel* to one of the following:

- **FINE**
- **FINER**
- **FINEST**
- **INFO** (This is the default.)
- **OFF**
- **SEVERE**
- **WARNING**

-v
Displays information about successful or failed processing as the utility runs.

-ow
Overwrites any existing pricing or setup components available in the PDC database with the specified components. If the specified pricing or setup component does not exist in PDC, the utility creates new pricing or setup components.

Copyright © 2017, 2026, Oracle and/or its affiliates.

> **Note**
>
> If you do not use the **-ow** parameter when importing data, the utility generates an error if an object with the same name already exists in PDC.

[-changeSetName *ChangeSetName*]
Specifies the name of the changeset used to import setup and pricing components into the PDC database. If you do not specify *ChangeSetName*, the utility generates the changeset name as **ieChangeSet***timestamp*, where *timestamp* is the current timestamp in the format *yyyy-mm-dd_hh-mm-ss*.

-appsvruser *ApplicationServerUserName*
Specifies the user name to access the WebLogic Server. If you do not specify this parameter, the utility prompts you to enter the user name at the command prompt.

-pdcuser *PdcUserName*
Specifies the user name to access PDC. If you do not specify this parameter, the utility prompts you to enter the user name at the command prompt.

-h|-help
Displays the syntax for this utility.

Publish Mode

The **publish** mode publishes setup components from PDC to the specified target engines, such as the batch rating engine, the real-time rating engine, and Oracle Communications Elastic Charging Engine (ECE).

publish Mode Syntax

```
ImportExportPricing -publish [metadata | config | pricing | all |
ObjectType1,ObjectType2,…]
                            -target [rre | bre | ece | all]
```

publish Mode Parameters

-publish
Publishes the specified components from PDC to the specified target engines.

metadata | config | pricing | all | *ObjectType1,ObjectType2,…*
Indicates the component types to publish. The default value is **all**.

- **metadata**: Publishes the metadata components in PDC to the specified target engine.
- **config**: Publishes the setup components in PDC to the specified target engine.
- **pricing**: Publishes the pricing components in PDC to the specified target engine.
- **all**: Publishes all components in PDC to the specified target engine.
- *ObjectType1,ObjectType2,…*: Publishes the specified components to the target engine; for example, BALANCE_ELEMENT. For the list of component types, see Table 39-3.

-target [rre | bre | ece | all]
Publishes the components or object types to the specified target engine:

- **rre**: Publishes the components or object types to the real-time rating target engine.

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **bre**: Publishes the components or object types to the batch rating target engine.
- **ece**: Publishes the components or object types to the ECE rating target engine.
- **all**: Publishes the components or object types to all target engines mentioned above.

If you do not specify a target engine, the utility publishes to all target engines.

Keep Mode

The **keep** mode retains the latest versions of successfully promoted PDC components, but deletes the older versions.

keep Mode Syntax

```
ImportExportPricing -keep [config | pricing | all | metadata] [-numVersion N]
```

keep Mode Parameters

-keep
Keeps only the latest versions of successfully promoted PDC components, and deletes older versions of the specified components that are obsolete in PDC or the associated target engines (for example ECE).

config | pricing | all | metadata
Indicates the component types affected. The default value is **all**.

- **config**: Keeps the latest versions of setup components, but deletes older versions that are obsolete.
- **pricing**: Keeps the latest versions of pricing components, but deletes older versions that are obsolete. See Table 39-3.
- **all**: Keeps the latest version of all PDC components.
- **metadata**: Keeps only the metadata components that are supported in your PDC system. By default, PDC supports the following metadata object types.

-numVersion *N*
Keeps the specified number of the most recent versions of all successfully promoted objects in the PDC database and deletes the remaining versions, where *N* is a positive integer that specifies the number of latest successful versions to be kept in the PDC database. If you do not specify a number, the utility keeps two by default.

Delete Mode

The **delete** mode deletes the specified setup components, as long as they are not used by other entities.

delete Mode Syntax

```
ImportExportPricing [-d | -delete] FileName
```

delete Mode Parameters

-d | -delete *FileName*
Deletes the setup components contained in *FileName* from the PDC database, where *FileName* is the XML file containing the setup components. The utility displays an error message if the data being deleted is dependent on other data.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Type Mode

The **type** mode displays the list of component types that are supported in your PDC system.

type Mode Syntax

```
ImportExportPricing [-type | -t] [config | pricing | brmObject | metadata]
```

type Mode Parameters

-type | -t
You can use **-type** and **-t** interchangeably. Displays the specified component types that are supported in your PDC system.

config | pricing | brmObject | metadata
Indicates the component types to include. If you do not specify a component type, the utility displays all components.

- **config**: Displays only the setup components that are supported in your PDC system.
- **pricing**: Displays only the pricing components that are supported in your PDC system. See Table 39-3.
- **brmObject**: Displays only the BRM objects that are supported in your PDC system.
- **metadata**: Displays only the metadata components that are supported in your PDC system. See Table 39-2.

Changeset Status Mode

Use **changeSetStatus** mode to check the current status of one or more changesets. You can check a changeset by its name or ID.

changeSetStatus Mode Syntax

```
ImportExportPricing [-changeSetStatus | -css] [-name changesetName | -n changesetName | -id changesetId | -inputReport]
```

changeSetStatus Mode Parameters

-changeSetStatus | -css
You can use **-changeSetStatus** and **-css** interchangeably. Retrieves the status of one or more changesets.

-name *changesetName* | -n *changesetName* | -id *changesetId*
Specifies the changesets to query.

- **-name** *changesetName*: Specifies one or more changeset names. To specify multiple changesets, separate the names with commas.
- **-n** *changesetName*: Equivalent to **-name**.
- **-id** *changesetId*: Specifies one or more changeset IDs. To specify multiple changesets, separate the IDs with commas.

-inputReport
Specifies the path to the changeset report file.

Copyright © 2017, 2026, Oracle and/or its affiliates.

PDCWalletUtil.sh

Use the **PDCWalletUtil.sh** utility to create the PDC wallet, set the passwords stored in the wallet, or retrieve the passwords stored in the wallet. For example, you can set or retrieve the passwords for the WebLogic Server domain, PDC user, and database.

For more information, see "Changing the Password in the PDC Wallet" in *BRM System Administrator's Guide*.

Location

PDC_home/**apps/bin**

Syntax Overview

You can run the **PDCWalletUtil.sh** utility in the following modes:

- **create**: Create the PDC wallet in the specified location and with the specified password. See "create Mode Syntax".
- **set**: Set or change a password in the specified configuration entry. See "set Mode Syntax".
- **get**: Retrieves the password from the specified configuration entry. See "get Mode Syntax".

create Mode Syntax

```
./PDCWalletUtil.sh create walletLocation
Enter wallet password: walletPassword
```

The **create** mode creates the PDC wallet in the specified location and with the specified password.

create Mode Parameters

walletLocation
The location in which to create the PDC wallet.

walletPassword
The PDC wallet password.

set Mode Syntax

```
./PDCWalletUtil.sh set walletLocation configEntry
Enter wallet password: walletPassword
Enter key password: keyPassword
Enter key password again: keyPassword
```

The **set** mode sets or changes the password in the specified configuration entry.

set Mode Parameters

walletLocation
The location in which the PDC wallet resides.

Copyright © 2017, 2026, Oracle and/or its affiliates.

configEntry
The configuration entry in which to store the password. The valid values are:

- **PDC_APP_SERVER_USER_PASSWORD**: The Oracle WebLogic Server user password.
- **PDC_APP_USER_PASSWORD**: The PDC user password.
- **TARGET_PDC_ADMIN_PASSWORD**: The password of the Oracle WebLogic Server user in the target system.
- **SOURCE_PDC_ADMIN_PASSWORD**: The password of the Oracle WebLogic Server user in the source system.
- **TARGET_PDC_USER_PASSWORD**: The password of the PDC user in the target system.
- **SOURCE_PDC_USER_PASSWORD**: The password of the PDC user in the source system.

walletPassword
The PDC wallet password.

keyPassword
The password to store in the configuration entry.

get Mode Syntax

```
./PDCWalletUtil.sh get walletLocation configEntry
Enter wallet password: walletPassword
```

The **get** mode retrieves the password from the specified configuration entry.

get Mode Parameters

walletLocation
The location in which the PDC wallet resides.

configEntry
The configuration entry from which to retrieve the password. The valid values are:

- **PDC_APP_SERVER_USER_PASSWORD**: The Oracle WebLogic Server user password.
- **PDC_APP_USER_PASSWORD**: The PDC user password.
- **TARGET_PDC_ADMIN_PASSWORD**: The password of the Oracle WebLogic Server user in the target system.
- **SOURCE_PDC_ADMIN_PASSWORD**: The password of the Oracle WebLogic Server user in the source system.
- **TARGET_PDC_USER_PASSWORD**: The password of the PDC user in the target system.
- **SOURCE_PDC_USER_PASSWORD**: The password of the PDC user in the source system.

walletPassword
The PDC wallet password.

Results

The **PDCWalletUtil.sh** utility notifies you when it successfully creates the wallet or stores the password in the PDC wallet.

Copyright © 2017, 2026, Oracle and/or its affiliates.

PDCBRMWalletUtil.sh

Use the **PDCBRMWalletUtil.sh** utility to create the BRM Integration Pack wallet and to set passwords stored in that wallet. For example, you can set passwords for the transformation cross-reference database, migration cross-reference database, and BRM database.

For more information, see "Changing Passwords in the Wallet" in *BRM System Administrator's Guide*.

Location

*BRM_Integration_Pack_home***/apps/bin**

Syntax Overview

You can run the **PDCBRMWalletUtil.sh** utility in the following modes:

- **create**: Create the BRM Integration Pack wallet in the specified location and with the specified password. See "create Mode Syntax".
- **set**: Set or change the password in the specified configuration entry. See "set Mode Syntax".

create Mode Syntax

```
./PDCBRMWalletUtil.sh create walletLocation
Enter wallet password: walletPassword
```

The **create** mode creates the BRM Integration Pack wallet in the specified location and with the specified password.

create Mode Parameters

walletLocation
The location in which to create the BRM Integration Packet wallet.

walletPassword
The password for the BRM Integration Pack wallet.

set Mode Syntax

```
./PDCBRMWalletUtil.sh set walletLocation configEntry
Enter wallet password: walletPassword
Enter key password: keyPassword
Enter key password again: keyPassword
```

The **set** mode sets or changes the password in the specified configuration entry.

set Mode Parameters

walletLocation
The location in which the BRM Integration Packet wallet resides.

configEntry
The configuration entry in which to store the password. The valid values are:

Copyright © 2017, 2026, Oracle and/or its affiliates.

- **MIGRATION_DB_PASSWORD**: The migration cross-reference database user password.
- **TRANS_XREF_DB_PASSWORD**: The transformation cross-reference database user password.
- **BRM_DB_PASSWORD**: The BRM database user password.

walletPassword
The password for the BRM Integration Pack.

keyPassword
The password to be stored in the configuration entry.

Results

The **PDCBRMWalletUtil.sh** utility notifies you when it successfully creates the wallet or stores the password in the BRM Integration Pack wallet.

load_pin_snowball_distribution

Use the **load_pin_snowball_distribution** utility to load snowball discount distribution rules into the **/config/snowball_distribution** object in the BRM database. You define how snowball discounts are distributed in the **pin_snowball_distribution** file in *BRM_home***/sys/data/pricing/example**.

> **Note**
>
> - The **load_pin_snowball_distribution** utility overwrites existing distribution rules. If you are updating distribution rules, you cannot load new distribution rules only. You must load complete sets of distribution rules each time you run the **load_pin_snowball_distribution** utility.
> - To connect to the BRM database, the **load_pin_snowball_distribution** 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_snowball_distribution [-d] [-v] [pin_snowball_distribution_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 data has not been loaded into the database.

-v
Displays information about successful or failed processing as the utility runs.

Copyright © 2017, 2026, Oracle and/or its affiliates.

> **Note**
>
> This parameter is always used in conjunction 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_snowball_distribution** *other_parameters* **-v** > *filename***.log**

pin_snowball_distribution_file
The name and location of the file that defines the snowball distribution rules. The default **pin_snowball_distribution** file is in *BRM_home***/sys/data/pricing/example**.
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_snowball_distribution BRM_home/sys/data/pricing/example
```

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 more information, see "Charging for Custom Events and Attributes" in *BRM Setting Up Pipeline Pricing*.

> **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 "Creating Configuration Files for BRM Utilities".

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.

Copyright © 2017, 2026, Oracle and/or its affiliates.

Results

If the 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_contributors

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 more information, see "Defining and Loading Custom Sub-Balances".

> **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*.

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.

Copyright © 2017, 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_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.

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.

pin_discount_cleanup

Use this utility to specify how to handle expired discounts:

- Change the status of active, expired discounts to canceled
- Change the status of inactive, expired discounts to canceled
- Delete expired discounts

You use this utility when you set discount midcycle purchase or cancel options that grant a full discount when the discount is canceled in the middle of an accounting cycle.

You can run this utility nightly or add it to the **pin_bill_day** script to be run automatically. See "Running Billing Utilities" in *BRM Configuring and Running Billing*.

Location

*BRM_home***/bin**

where *BRM_home* is the directory in which the BRM server software is installed.

Syntax

```
pin_discount_cleanup -m close|close_inactive|delete [-n days] [-d date] [-v] [-t] [-help]
```


Copyright © 2017, 2026, Oracle and/or its affiliates.

Parameters

-m close | close_inactive | delete
Specifies how to handle discounts that expire on the specified date. The default date is the current date.

- **close**

  Changes the status of all active, expired discounts to canceled without deleting the discounts.

- **close_inactive**

  Changes the status of all inactive, expired discounts to canceled without deleting the discounts.

- **delete**

  Deletes all expired discounts.

-n *days*
The number of days prior to **-d** *date* for which to cancel or delete expired discounts. This is used in conjunction with the **-d** *date* parameter to determine the actual dates for canceling or deleting expired discounts. The default is 0.

-d *date*
The end date (in the format *MM/DD/YYYY*) for which to cancel or delete expired discounts. This is used in conjunction with the **-n** *days* parameter to determine the actual dates for canceling or deleting discounts. The default is the current date.
For example, if you enter **-n 5** and **-d 07/15/2021**, the utility cancels or deletes all discounts that expired on 7/10/2021.

> **Note**
>
> - The expire date cannot be greater than the current date. For instance, in the example above, if 7/10/2021 is greater than the current date, **pin_discount_cleanup** returns an error.
> - If neither the -**n** or **-d** parameters are specified, the current date is used.

-v
Displays information about successful or failed processing as the utility runs.

> **Note**
>
> This parameter is always used in conjunction 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_discount_cleanup** *other_parameters* **–v** > *filename*.**log**

Copyright © 2017, 2026, Oracle and/or its affiliates.

-t
Displays the number of records processed (the number of expired discounts that were canceled or deleted).

-help
Displays the syntax and parameters for this utility.

Results

To check the results of running this utility, look in the log file (normally **default.pinlog**) for error messages. The log file is located in the directory from which the utility was started or in a directory specified in the utility's configuration file (**pin.conf**).

Copyright © 2017, 2026, Oracle and/or its affiliates.

A

New Pricing Terminology

Learn about the pricing terminology changes for this release of Oracle Communications Billing and Revenue Management (BRM) and Pricing Design Center (PDC).

Topics in this document:

- New Pricing Terminology

New Pricing Terminology

Pricing terminology changed in this BRM release. Table A-1 lists those differences.

Table A-1 Pricing Terminology

Previous Term	New Term
ChargeShare	chargeshare offer
ChargeShare model	chargeshare
deal	bundle
discount	discount offer
discount master	discount filter
discount model	discount
EVAL("*function_name*") discount expression	Function["*function_name*"] discount expression
plan	package
plan list	package list
product	charge offer
rate plan	charge
rate plan selector	charge selector
resource	balance element (design time) or balance (runtime)
subscription group	service group

Copyright © 2017, 2026, Oracle and/or its affiliates.