

**Oracle® Communications
Billing and Revenue Management**

Setting Up Pricing and Rating

Release 7.5

E16711-18

December 2019

Copyright © 2011, 2019, Oracle and/or its affiliates. All rights reserved.

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 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 installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. 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 and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon 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, Opteron, the AMD logo, and the AMD Opteron 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.

Preface

This document describes various aspects of pricing, including setting up a price list, and rating, including using real-time rating or Pipeline Manager. This document also covers rerating.

Audience

This document is intended for system administrators and other professionals involved in pricing, rating, and rerating.

Accessing Oracle Communications Documentation

BRM documentation and additional Oracle documentation; such as Oracle Database documentation, is available from Oracle Help Center:

<http://docs.oracle.com>

Additional Oracle Communications documentation is available from the Oracle software delivery Web site:

<https://edelivery.oracle.com>

Documentation Accessibility

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>.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit

<http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit

<http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

Document Revision History

The following table lists the revision history for this book.

Version	Date	Description
E16711-01	November 2011	Initial release.

Version	Date	Description
E16711-02	May 2012	Documentation updates for Oracle Communications Billing and Revenue Management (BRM) 7.5 Patch Set 1. <ul style="list-style-type: none"> ■ Made minor formatting and text changes.
E16711-03	August 2012	Documentation updates for BRM 7.5 Patch Set 2. <ul style="list-style-type: none"> ■ Added documentation for the "load_brm_pricing" utility.
E16711-04	December 2012	Documentation updates for BRM 7.5 Patch Set 3. <ul style="list-style-type: none"> ■ Added "Policy-Driven Charging" to document the use of offer profiles in policy-driven charging. ■ Updated "loadpricelist" to document the inclusion of offer profiles data. ■ Added "Specifying Which Non-Currency Sub-Balances to Load on Startup" to document how DAT_BalanceBatch loads non-currency sub-balances.
E16711-05	March 2013	Documentation updates for BRM 7.5 Patch Set 4. <ul style="list-style-type: none"> ■ Added "Configuring Rerating for Accounts Associated With Subscription Service Transfer". ■ Replaced multidatabase information with multischema information.
E16711-06	July 2013	Documentation updates for BRM 7.5 Patch Set 5. <ul style="list-style-type: none"> ■ Added "Configuring BRM for Elastic Charging Engine Rerating".
E16711-07	August 2013	On HP-UX IA64, BRM 7.5 is certified as of BRM 7.5 Patch Set 5. Documentation added for HP-UX IA64.
E16711-08	October 2013	Documentation updates for BRM 7.5 Patch Set 6. <ul style="list-style-type: none"> ■ Updated "Loading the ECE Payload Configuration File". ■ Added the "Configuring BRM for Elastic Charging Engine Rerating" section.
E16711-09	February 2014	Documentation updates for BRM 7.5 Patch Set 7. <ul style="list-style-type: none"> ■ Made minor formatting and text changes.
E16711-10	August 2014	Documentation updates for BRM 7.5 Patch Set 9. <ul style="list-style-type: none"> ■ Documentation added for RADIUS Manager.
E16711-11	October 2014	Documentation updates for BRM 7.5 Patch Set 10. <ul style="list-style-type: none"> ■ Made minor formatting and text changes.

Version	Date	Description
E16711-12	January 2015	Documentation updates for BRM 7.5 Patch Set 11. <ul style="list-style-type: none"> Added "Enabling Calculation of Deferred Taxes During Rerating".
E16711-13	August 2015	Documentation updates for BRM 7.5 Maintenance Patch Set 1. <ul style="list-style-type: none"> Replaced the phrase "BRM CDR format" with the phrase "Oracle CDR format". Added the "About Rerating Events by Using the Rates Applied when the Rating Conditions Change During the Session" section. Updated the following sections: <ul style="list-style-type: none"> Prerequisites for Creating a Price List About Real-Time Rate Plans
E16711-14	December 2015	Documentation updates for BRM 7.5 Patch Set 14. <ul style="list-style-type: none"> Added the following sections: <ul style="list-style-type: none"> About Applying Folds Only For Account-Level Products Configuring Time-Stamp Rounding for Purchase Grants Updated the following sections: <ul style="list-style-type: none"> Deleting Unused Free Hours Creating Friends and Family ERAs Using the Correct Time Zone About Using Filters for Loading Events About Balance Impacts That Become Valid on First Usage Using Event Attributes to Define Impact Categories
E16711-15	April 2016	Documentation updates for BRM 7.5 Patch Set 15. <ul style="list-style-type: none"> Made minor formatting and text changes.

Version	Date	Description
E16711-16	August 2016	<p>Documentation updates for BRM 7.5 Patch Set 16.</p> <ul style="list-style-type: none"> ■ Updated the following sections: <ul style="list-style-type: none"> Specifying the Rating Mode Specifying Events for Rerating About the Real-Time Rerating Pipeline How Comprehensive Rerating Works How BRM Applies Rounding Modifying Rated Events ■ Added a note in the following tables: <ul style="list-style-type: none"> Creating the Monthly Internet Fee Creating the Monthly Email Fee Defining the Product Creating the Standard Sign-up Fee Creating the Free Rate
E16711-17	December 2016	<p>Documentation updates for BRM 7.5 Patch Set 17.</p> <ul style="list-style-type: none"> ■ Updated the following sections: <ul style="list-style-type: none"> Specifying Events for Rerating Parameters for Selecting Accounts for Rerating
E16711-18	December 2019	<p>Documentation updates for BRM 7.5 Patch Set 23.</p> <ul style="list-style-type: none"> ■ Updated the following sections: <ul style="list-style-type: none"> Configuring Offer Profile Data Managing Offer Profile Data Managing Offer Profiles with Opcodes loadpricelist

Part I

Pricing Concepts

Part I describes price list concepts, including plans and deals. To help you apply the concepts more quickly, it describes the sample pricing plans included in Oracle Communications Billing and Revenue Management (BRM).

Part I contains the following chapters:

- [About Creating a Price List](#)
- [Understanding the Sample Pricing Plans](#)

About Creating a Price List

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

Before reading this chapter, you should have a basic understanding of BRM concepts. See *BRM Concepts*.

About Price Lists

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

Plans

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

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

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

Deals

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

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

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

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

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

Products

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

Your products might include:

- A setup fee.
- A monthly subscription fee.
- Usage fees for telephone calls.
Usage fees can be rated in real time or in a batch rating pipeline.
- An offer profile
An offer profile is associated with a product or discount.

Offer Profiles

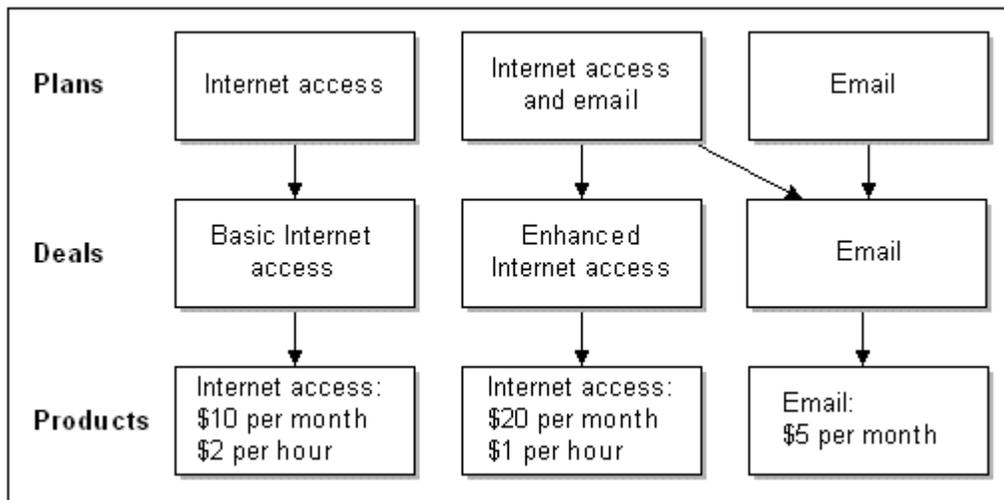
You use *offer profiles* in association with products and discounts to provide policy-driven provisioning of services. Each offer profile consists of the following data:

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

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

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

Figure 1–1 Simple Price List



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

What Is Rating?

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

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

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

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

About Billable Events

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

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

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

How BRM Rates a Billable Event

You can rate billable events in two ways:

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

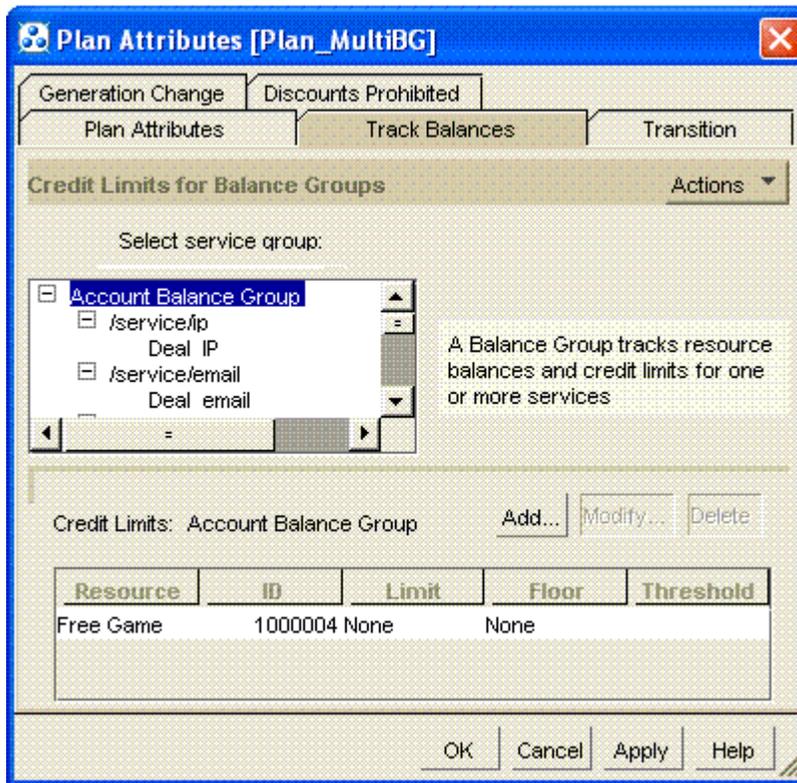
About Real-Time Rating

Real-time rating is used for services such as dialup access and email. For example, to rate Internet access in real time:

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

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

Figure 1–2 Track Balances Tab



About Pipeline Batch Rating

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

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

Note: You can also use UE Loader to load data from event log files as billable events and rate them using real-time rating. The event log files can include data from Web servers or other services that record events in files. When the events are loaded, BRM rates the events and updates the customers' account balances. For more information, see "[About Rating Events Created by External Sources](#)".

For more information, see "About pipeline rating" in *BRM Configuring Pipeline Rating and Discounting*.

About Different Types of Rates

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

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

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

Important: You use pipeline rating only for rating usage events. You can use real-time rating to rate all types of events.

About External and Internal Events

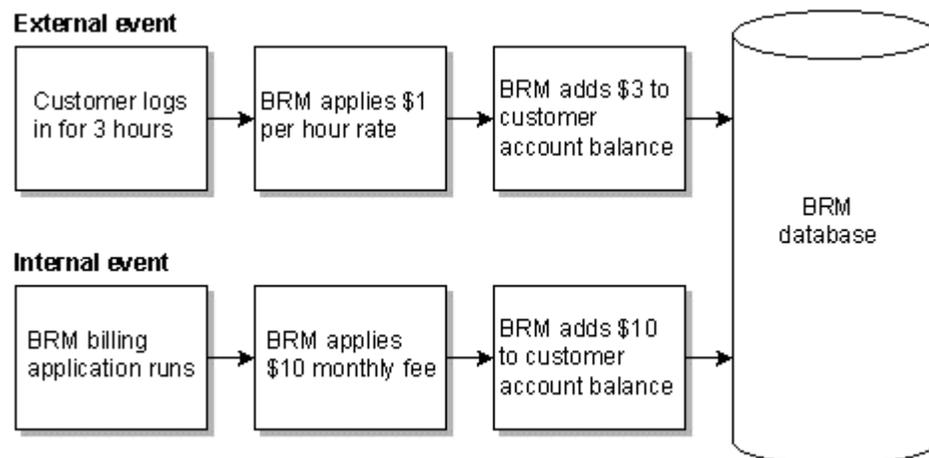
An *external* billable event is a usage event created by the customer outside of the BRM system (for example, when a customer makes a telephone call).

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

Different rates are used to rate external and internal events.

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

Figure 1-3 External and Internal Event Rating Flow



About Cycle Rates

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

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

About Cycle Forward Events

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

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

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

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

About Cycle Arrears Events

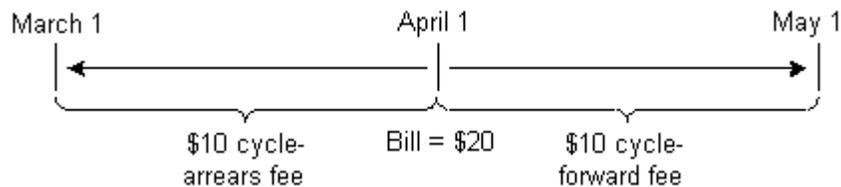
Cycle arrears events occur at the end of the month to charge the customer for the past month. When a customer pays a cycle arrears fee, the customer pays for the month that has already occurred.

Note: There are no multi-month cycle arrears fees.

In [Figure 1–4](#), the customer owns two plans, each with a cycle fee. The bill created on April 1 includes:

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

Figure 1–4 *Cycle Arrears and Cycle Forward Fees*



About Cycle Forward Arrears Events

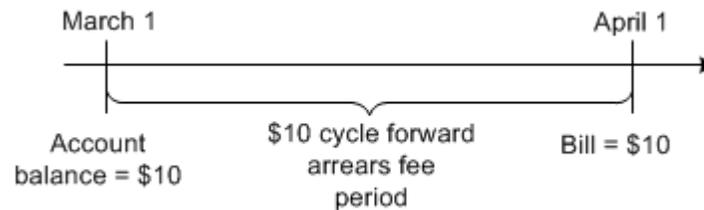
Cycle forward arrears events occur at the beginning of the month to charge customers for the upcoming month, but the cycle fees are not billed until the end of the month when billing is run. When a customer pays a cycle forward arrears fee, the customer pays for the month that has just passed.

Note: There are no multi-month cycle arrears fees.

BRM adds cycle forward arrears fees to account balances as unbilled revenue at the beginning of the cycle. This enables you to recognize unbilled cycle arrears fees when you run G/L reports before the cycle ends. For more information, see "About collecting general ledger data" in *BRM Collecting General Ledger Data*.

Cycle forward arrears fees are stored in cycle forward arrears items. The cycle forward arrears event is assigned to 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 as shown in [Figure 1-5](#):

Figure 1-5 Cycle Forward Arrears Fee



About using delayed billing

Because cycle forward arrears fees are assigned to the item that belongs to the next cycle, you must use delayed billing even if you do not need a delayed billing period.

When you use delayed billing, BRM tracks events in the current cycle that will be billed in the next accounting cycle. When the bill item for the next accounting cycle is created, BRM then assigns the tracked events to that bill item.

If you do not need a delayed billing period, you can set the value of the delayed period to 0.

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

How BRM Creates Cycle Events

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

For information about accounting cycles, see "About accounting and billing cycles" in *BRM Configuring and Running Billing*.

Benefits of Using the Cycle Event Types

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

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

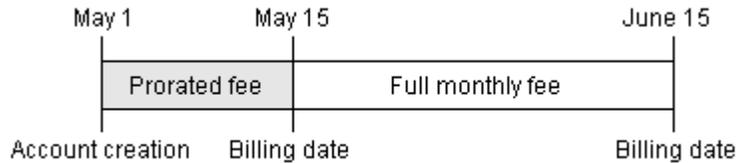
The advantage of using a cycle forward arrears event is that all of the usage and cycle fees for a particular month are included in the same bill, *and* the cycle fee is recorded in the G/L when a G/L report is run before the fee is billed.

Allowing Cycle Fees to Be Prorated

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

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

Figure 1–6.

Figure 1–6 Prorating Cycle Fees

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

- When a customer changes the billing date and a partial month is created as a result.
- When a customer cancels a product before the end of the billing cycle.

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

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

For details on proration, see "Calculating prorated cycle fees" in *BRM Configuring and Running Billing*.

Applying Multiple Rates to the Same Event

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

- You can apply different time of day rates to a single type of dialup event. See:
 - [Real-Time Rating Based on Date and Time](#)
 - [Rating by Date and Time with Pipeline Manager](#)
- You can apply a different rate based on the quantity that has already been rated. See "[Real-Time Rating Based on Event or Resource Quantity](#)".

Ways to Rate Events

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

About Ratable Usage Metrics

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

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

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

- Pipeline batch rating: "[Setting Up Pipeline Price List Data](#)".
- Real-time rating: "[About Setting Up Rums for Real-Time Rating](#)".

About Applying Multiple RUMs to Rate an Event

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

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

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

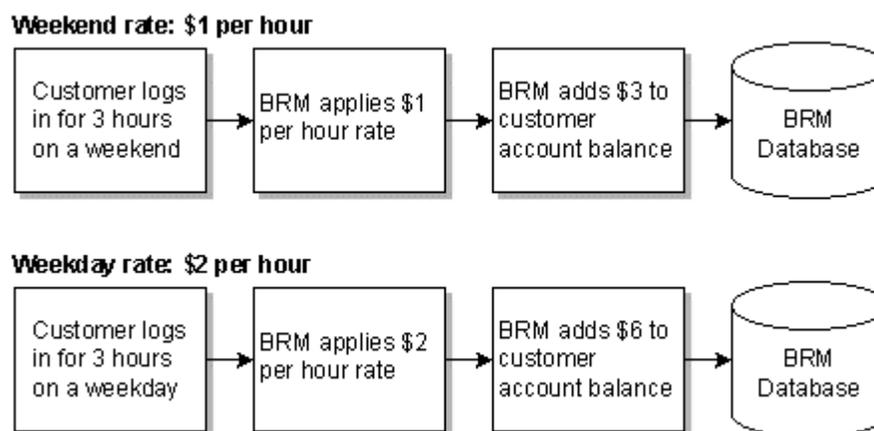
For more information, see "[Real-Time Rating Based on Multiple RUMs](#)".

About Rating Based on Date and Time

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

[Figure 1-7](#) shows how the same event can be rated differently based on when it occurred:

Figure 1-7 Rating Based on Date and Time



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

- Pipeline batch rating: "[Rating by Date and Time with Pipeline Manager](#)".
- Real-time rating: "[Real-Time Rating Based on Date and Time](#)".

About Resources

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

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

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

- Pipeline batch rating: ["Rating by Date and Time with Pipeline Manager"](#).
- Real-time rating: ["Real-Time Rating Based on Date and Time"](#).

About Rounding Resources

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

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

You set up rounding rules for pipeline batch rating and real-time rating. For information, see ["About Resource Rounding"](#).

About Setting Rules for Resource Consumption

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

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

About Rating Based on Event Attributes

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

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

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

- Pipeline batch rating: ["Rating by Date and Time with Pipeline Manager"](#).
- Real-time rating: ["Real-Time Rating Based on Date and Time"](#).

About Rating Based on Event or Resource Quantity

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

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

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

- Pipeline batch rating: "[Rating by Date and Time with Pipeline Manager](#)".
- Real-time rating: "[Real-Time Rating Based on Date and Time](#)".

About Products

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

Specifying Rates in Products

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

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

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

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

For example, a product for rating an IP Dialup service might include the rating information shown in [Table 1-1](#):

Table 1–1 Rating Information for Dialup Service

Event: Monthly Cycle Forward Event	Event: IP Dialup Event
<ul style="list-style-type: none"> ■ RUM: Occurrence ■ Resource: US Dollars ■ Balance impact: \$20 per occurrence 	<ul style="list-style-type: none"> ■ RUM: Duration ■ Resource: US Dollars ■ Balance impact: \$1 per hour

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

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

Using products to rate different services

When you offer multiple services, you typically create products for each service. [Table 1–2](#) shows an example:

Table 1–2 Products and Services

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

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

Using products to charge different amounts for the same service

You can use products to charge for the same service in different ways. You can do this by rating combinations of internal and external events. [Table 1–3](#) shows an example:

Table 1–3 Rating Combinations

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

Associating products with Offer Profiles

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

Associating products with accounts

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

Table 1–4 Associating Products and Accounts

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

Using products to handle complex rating

You can rate a single event in different ways in the same product. [Table 1–5](#) shows an example:

Table 1–5 Complex Rating with Products

Simple Product	Complex Product
\$10 purchase fee	Purchase fees: <ul style="list-style-type: none"> ▪ \$10 purchase fee if purchased after December 2007 ▪ \$5 purchase fee if purchased after October 2007 ▪ No purchase fee if purchased before or during October 2007
\$1 per hour usage fee	Usage fees: <ul style="list-style-type: none"> ▪ \$1 per hour on weekends ▪ \$2 per hour on weekdays ▪ First 20 weekend hours free ▪ First 10 weekday hours free

Specifying Minimum Event Quantities for Rate Plans

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

Specifying How to Round Event Quantities

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

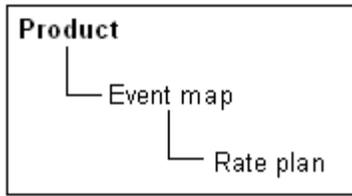
You can take advantage of event quantity rounding to charge for consistent blocks of time (for example, 5 cents for each 10 seconds of an IP telephony call).

Note: Event quantity rounding is different from resource rounding. With resource rounding, you round the amount to charge. With event quantity rounding, you round the amount of usage. See "[Setting Up Resources for Real-Time Rating](#)".

About Organizing Rates in a Product

You organize rates in a product in a hierarchical structure as shown in [Figure 1–8](#):

Figure 1–8 Organizing Rates in a Product



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

About the Event Map

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

- Monthly Cycle Forward Event
- IP Dialup Event

See "[Mapping Event Types to Services](#)".

About Rate Plans

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

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

About Product Ownership

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

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

For example, you might have two customer accounts that use the same service, but have different products as shown in [Table 1–6](#):

Table 1–6 Products and Accounts

Account 1	Account 2
IP product 1: <ul style="list-style-type: none"> ■ Service: IP Dialup ■ Cycle forward event: \$20 ■ Usage events: \$2 	IP product 2: <ul style="list-style-type: none"> ■ Service: IP Dialup ■ Cycle forward event: \$10 ■ Usage events: \$1

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

Note: When you manage a customer's account, you manage their products. For example, you can change the product status or customize the product's pricing. When you create your price list, it is important to consider how the products will be managed after they have been purchased.

Specifying Which Services Are Rated by a Product

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

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

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

- Account-level product: \$20 monthly fee
- IP Dialup service product: \$10 per occurrence

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

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

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

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

About Specifying the Events to Rate in a Product

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

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

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

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

How you choose which events are rated in a product depends in large part on the services you offer, but it also depends on how you manage your business and your

customers. For example, you might change your pricing often, provide sponsored services, or provide numerous types of discounts. Before you finalize your price plan, run some discounting and customer management scenarios to determine if your products are organized in a way that supports your business. For more information, see the following topics:

- "Managing customers' products and services" in *BRM Managing Customers*.
- "About sponsor groups" in *BRM Managing Accounts Receivable*.

Specifying Product Types

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

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

You specify product types when you use Pricing Center to create a product as shown in [Figure 1-9](#):

Figure 1-9 Product Type in Pricing Center

The screenshot shows a form titled "Product Type". It contains three radio buttons: "Item", "Subscription", and "System". The "Subscription" radio button is selected. To the right of the "Subscription" radio button is the text "Applies to:" followed by a text input field containing "/service/ip" and a dropdown arrow.

Specifying General Product Information

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

Defining How Many Products a Customer Can Purchase

You can define the following quantity attributes:

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

Restricting When a Product Is Available

You can specify that a product is always valid, valid from a future date forward, valid until a future date, or valid for some definite period in the future as shown in [Figure 1-10](#).

Figure 1-10 Restricting Product Availability

The screenshot shows a 'Product Period' configuration window. It is divided into two main sections: 'Start' and 'End'.
 In the 'Start' section, there are two radio button options: 'Immediately' (which is selected) and 'At 12:00:00 AM On 09-Mar-2006'.
 In the 'End' section, there are two radio button options: 'Never' (which is selected) and 'At 12:00:00 AM On 09-Mar-2006'.

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

You can also specify:

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

Specifying Product Priority

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

For information on enabling product priority while applying cycle fee, see "Enabling Product Priority While Applying Cycle Fee" in *BRM Configuring and Running Billing*.

Rating Based on Product Provisioning

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

For more information, see ["About Using Product-Level Provisioning to Configure Services"](#).

Providing Deal-Level Discounts

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

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

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

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

$[(\text{rate} * \text{quantity}) - \text{discount}]$

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

$[(\text{rate} - \text{discount}) * \text{quantity}]$

See "[Setting Optional Rating Flags](#)" for instructions about setting this option.

About Deals

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

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

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

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

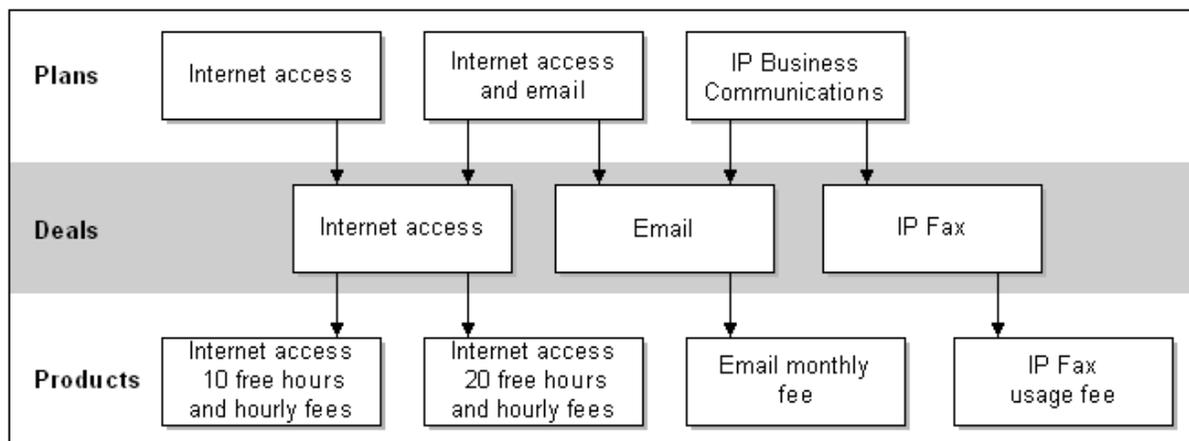
Each deal is typically associated with a specific service. For example:

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

One deal can contain any number of products, and two different deals can contain the same product. Therefore, packaging products in deals adds flexibility to your pricing structure without requiring you to create additional products. [Figure 1-11](#) shows a sample offering consisting of related plans, deals and products.

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

Figure 1–11 Deal Relationships to Plans and Products



About the Best Pricing Configuration

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

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

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

About Base Deals

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

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

Important: You can configure only one base deal in each best pricing configuration.

About Alternate Deals

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

Important: The number of alternate deals in your configuration affects performance of the best pricing calculation.

For more information, see "Offering the best price to your customers" in *BRM Configuring and Running Billing*.

For each alternate deal, you can specify:

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

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

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

An Example of Best Pricing Calculation

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

Table 1-7 Best Pricing Calculation Example

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

The charges calculated are shown in [Table 1-8](#):

Table 1-8 Calculating Charges

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

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

Purchasing Deals after an Account Is Created

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

Providing Discounts with Deals

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

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

Note:

- You can specify if rates are discountable. See "[Providing Deal-Level Discounts](#)".
 - You can provide fractional discounts up to two decimal places (for example, 10.25%, but not 10.255%).
-
-

Prohibiting, Allowing, and Requiring Deal Modification

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

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

About Product and Discount Validity Periods

Products and discounts have the following validity periods:

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

Setting the Effective Periods of Products and Discounts

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

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

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

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

- **Immediately:** The effective period starts as soon as the customer purchases the product or discount.
- **On first usage:** The effective period starts when the product or discount is first used. The product fees are not charged until the customer uses a service in the product for the first time: for example, by making a phone call. For more information, see "[About Effective Periods That Start on First Usage](#)".
- **Relative to the product or discount purchase time:** The purchase time is the time when a product or discount is added to the account.
 - When the purchase period has a relative start time, the purchase fee is charged when the product is purchased, but the customer cannot use the product's service or benefit from the discount until the relative period ends.
 - When a product's cycle or usage period has a relative start time, the cycle or usage fees are not charged (or rated) until the relative period ends, even if the service has been activated. This option enables you to waive subscription or usage fees for a period.
 - When a discount's cycle or usage period has a relative start time, the discount is not applied to cycle or usage fees until the relative period ends.

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

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

About Effective Periods That Start on First Usage

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

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

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

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

About activating first-usage discounts

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

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

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

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

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

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

About setting first-usage validity during pipeline rating

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

- Batch rating pipeline. See "About rating with products and discounts whose validity starts on first usage" in *BRM Configuring Pipeline Rating and Discounting*.
- Batch discounting pipeline. See "About setting the validity of resources impacted by discounts for more information" in *BRM Configuring Pipeline Rating and Discounting*.
- Real-time rerating pipeline. See "Configuring rerating to reset first-usage validity periods" in *BRM Setting Up Pricing and Rating*.

About Product and Discount Status at Purchase

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

You define reason codes in the reasons.locale file. For more information, see "About the reasons.locale file" in *BRM Configuring and Collecting Payments*.

Assigning Services to Deals

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

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

Using Deals to Bill Customers on Demand

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

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

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

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

About Deal Dependencies

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

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

Note: A prerequisite can contain deals of different services. For example, to own a GPRS deal, an account must own a GSM deal.

Note: A prerequisite deal cannot contain item products. When you create a plan with alternate deals, and if the base deal contains an item product, the purchase fails.

- **Plan requirements.** Specifies whether deals are optional or required for plans. Required deals must be purchased when a plan is purchased, whereas optional deals can be added at any time.

- **Mutual exclusivity.** Sets up a mutually exclusive relationship between two deals so if an account owns one deal, it cannot own the other.
- **Allowed transitions.** Specifies which deals or plans can serve as replacements for others.

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

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

Strategies for Creating Deals

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

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

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

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

Using Deals across Time Zones

When a deal is created, the start and end dates are set to the current date at midnight GMT (Greenwich Mean Time). For example, if a user in the US creates a deal at any time on January 15th, the system stores it as January 15th, midnight GMT. If the BRM server is located in London, its time is set to GMT/London time, which is 5 hours later than the US-based user's computer. If the US-based user opens a deal using **Modify Product**, the deal's start time is posted as 5 hours earlier than January 15th/midnight, or 7:00 PM on January 14th. The start date is then recorded as January 14th, rather than January 15th.

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

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

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

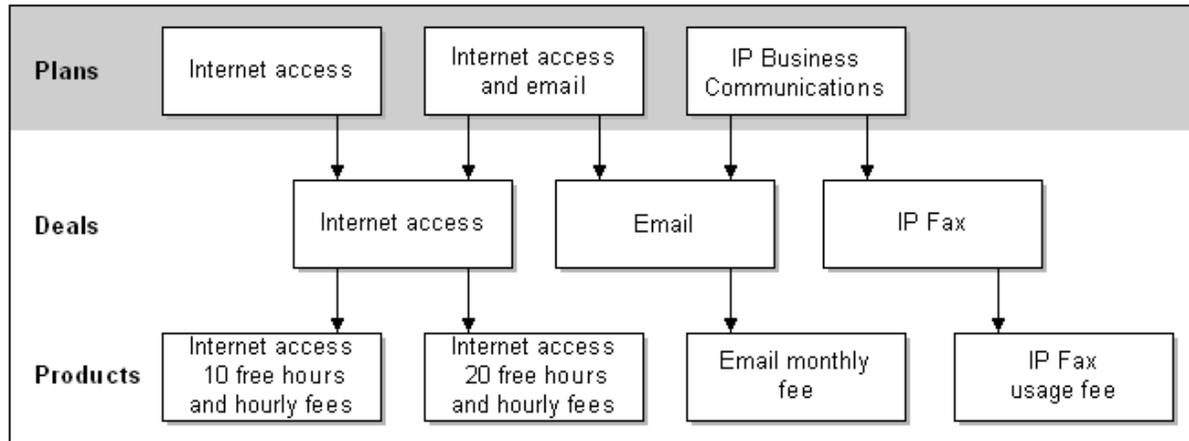
About Plans

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

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

Figure 1–12 shows how you can include deals, and the services they apply to, in a variety of plans.

Figure 1–12 Plan Relationships to Deals and Products



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

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

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

About Applying Credit Limits to Resources

You use plans to apply credit limits to resources.

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

Note: The default credit limit for currency resource is NULL and non-currency resource is 0. The default currency resource must have a positive value and the non-currency resource must have a negative value.

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

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

You use Customer Center to change the credit limit.

Though the credit limits are defined in the plans that customers purchase, the same credit limits must be defined in the input list when the customer account is created. The following scenarios apply:

About Credit Thresholds and Credit Floors

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

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

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

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

For non-currency resources, such as prepaid hours, you can use a negative number for the credit floor. For example, if you give 100 prepaid hours and you set the credit limit to 0; when the credit limit is reached, the customer has no hours remaining and cannot use the service. To notify the customer when there are only 10 hours left, you set the credit threshold and floor as follows:

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

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

Note: You can adjust a customer's credit limit.

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

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

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

Credit Limit and Floor Options

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

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

- You can choose whether to impose a zero credit limit if there is a balance impact to a resource in an account where the balance element for that resource is missing. If you choose to use a zero credit limit, the balance for that resource cannot exceed zero. If you choose not to use the credit limit, the balance can be any amount.

You set these options by changing entries in the Connection Manager (CM) configuration file (*BRM_Home/sys/cm/pin.conf*). See ["Setting Optional Rating Flags"](#) for details.

Tracking Resources by Service

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

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

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

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

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

Grouping Services by Subscription

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

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

For more information, see "Managing customers' subscription-level services" in *BRM Managing Customers*.

Creating CSR Plans

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

To create a CSR plan:

1. Use Pricing Center to create a plan that has the following attributes:
 - Use the **admin_client** service. This service provides access to Customer Center users.

- Include no deals.
2. Add the CSR plan to the **CSR - new** plan list.

CSR plans serve two purposes:

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

Using Plans to Bill Customers on Demand

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

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

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

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

Strategies for Creating Plans

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

Important: A plan name can include a maximum of 255 characters. A plan description can include a maximum of 1023 characters.

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

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

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

Transitioning between Plans

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

For generation change, you can transition customers between 2G (second generation) and 3G (third generation) wireless plans and services. Plans are called 2G or 3G depending on whether they have a second- or third-generation service as their primary service type.

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

For more information about defining generation change transitions, see "Defining a generation change for plans" in *BRM Managing Customers*.

About Plan Lists

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

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

Grouping plans into plan lists enables you to:

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

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

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

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

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

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

BRM includes two types of plan lists:

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

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

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

About the Default and CSR Plan Lists

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

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

Note: The **webclient** name is the default but can be changed. See "Specifying which plan list to display" in *BRM Managing Customers*.

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

About Offer Profiles

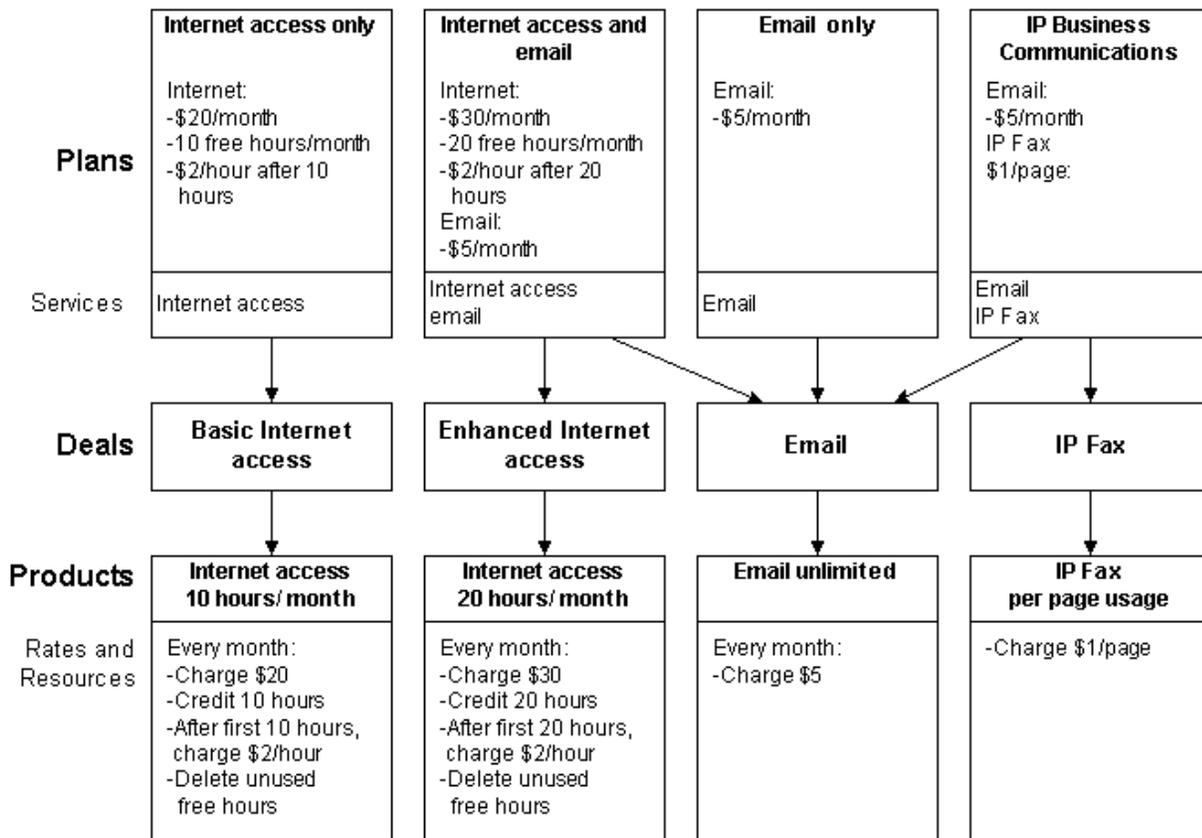
Offer profiles are made up of one or more policy labels each of which defines a gradation in the quality of service (QoS) based on usage amounts for a resource.

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

Price List Example

In the sample price list shown in [Figure 1-13](#), three services are offered: Internet access, email, and IP fax service. All rate plans used in this price list are real-time rate plans.

Figure 1–13 Sample Price List



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

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

For information about the sample plans, see "[Understanding the Sample Pricing Plans](#)".

About Setting Up a Price List

Setting up a price list typically requires the following steps:

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

it implements the company's pricing model. See ["Planning Your Price List"](#).

4. An operations person uses Pricing Center to create the price list. Because products are the basic unit of the price list, they are created first. See ["About Using Pricing Center"](#).
5. An operations person tests the price list. See ["Testing Your Price List"](#).

Prerequisites for Creating a Price List

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

- **Create services and events**

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

- **Create resources**

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

- **Define currency conversion rates**

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

- **Create G/L IDs**

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

- **Define tax codes and tax suppliers**

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

- **Set up offer profiles**

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

- **Define product-level provisioning categories**

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

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

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

- **Map event types to services**

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

- **Define zones**

For real-time rating, you use zones to create a single value to represent a group of values. You use the representative value in a rate plan selector. See ["About Using Zones to Group Event Attributes"](#).

For information on zones in batch pipeline rating, see ["Setting Up Zones for Batch Pipeline Rating"](#).

- **Define impact categories**

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

For pipeline rating impact category information, see ["About Impact Categories"](#).

- **Define pipeline data**

If you use pipeline rating, you must define several types of data and pricing components. See ["Setting Up Pipeline Price List Data"](#) and ["Data Required to Create Rate Plans"](#).

Planning Your Price List

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

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

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

About Using Pricing Center

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

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

With Pricing Center, you can perform the following tasks:

- Create and modify products, deals, plans, and plan lists.
- Create and modify rate plans for products. You can create rate plans for real-time rating and for pipeline batch rating.

- Define complex rate plans for your services based on valid time periods, event quantities, and rate priorities.
- Set credit limits and discounts.
- Reuse deals, plans, products, and rate plans so you do not need to re-create them every time.
- Update your pricing model without stopping and restarting your BRM system.
- Save a pricing document to a binary file and open the file at a later date. This enables multiple people to work on different pricing objects that will be committed to the same database.

Who Uses Pricing Center?

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

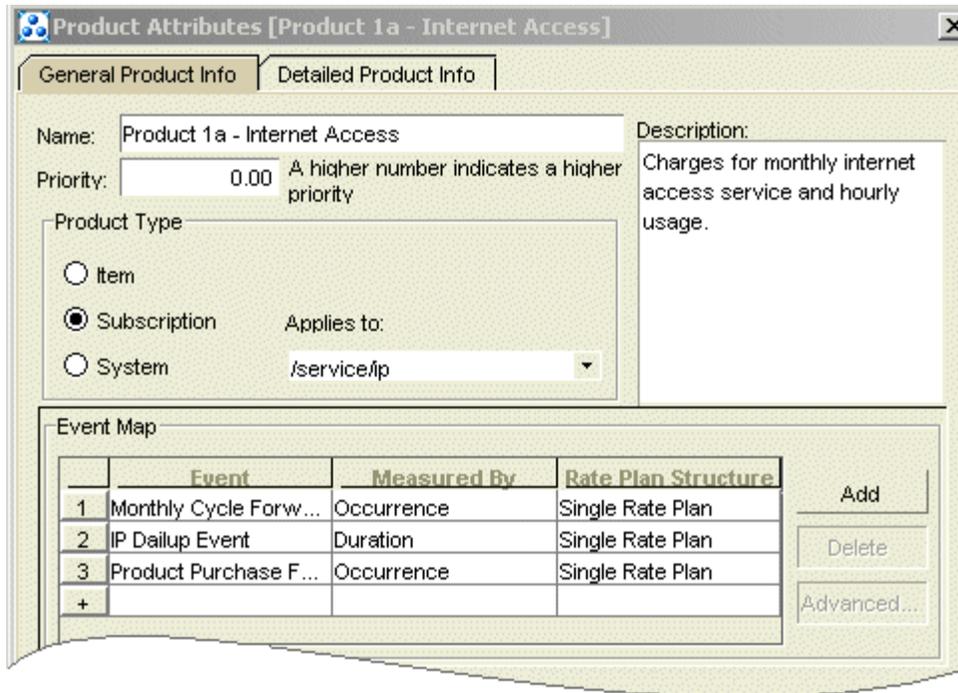
Example of Using Pricing Center

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

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

1. Start by using the Product Creation wizard to create the products. The wizard steps you through the general product parameters, such as the name and description of the product, the service it applies to, and the start and end dates.
2. When you finish, Pricing Center displays the product attributes that you selected. You can change them at any time.
3. After the general product properties are defined, you define the rate plans that specify how much to charge. In the example illustrated in [Figure 1-14](#), the product includes an IP connection fee, a monthly fee, and a purchase fee, so you create rate plans for those events:

Figure 1–14 Product Example



- For real-time rate plans, you apply balance impacts to each rate plan. [Figure 1–15](#) shows the balance impacts for a purchase fee:

Figure 1–15 Balance Impact Example



- When the product is finished, you create a deal and add the product as shown in [Figure 1–16](#):

Figure 1–16 Adding Product to Deal Example

Deal Attributes [Deal 1a - Measured Internet Access]

Name: Deal 1a - Measured Internet Access

Description:

Applies to: /service/tp

Deal customization: Optional

Bill on demand:

Valid Deal Period

Start: Immediately

End: Never

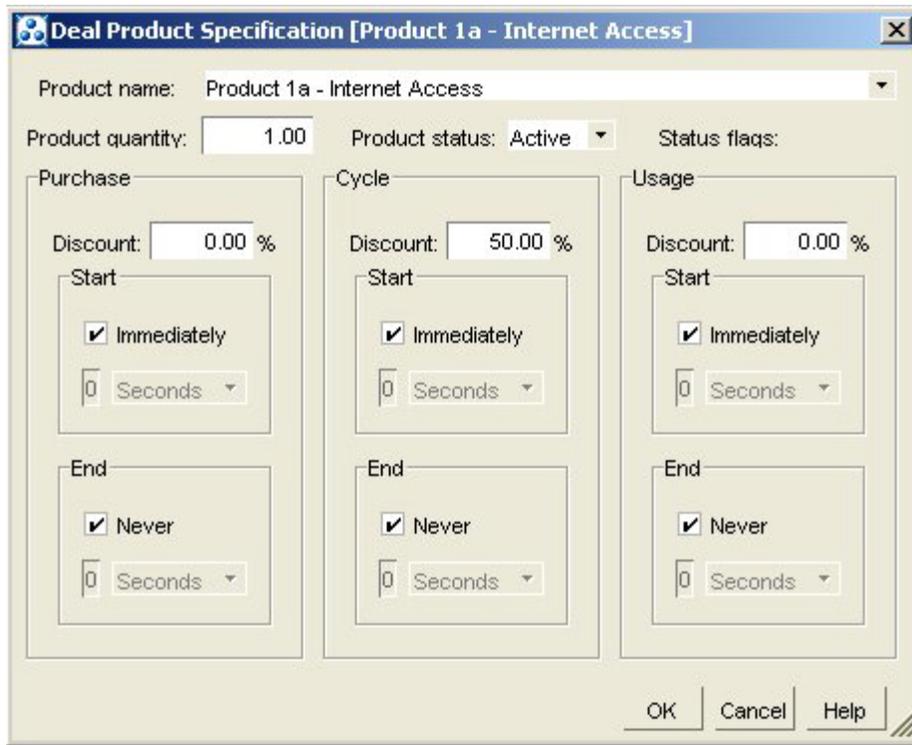
Products Associated with Deal

Product	Quantity
Product 1a - Internet Access	1.00

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

To discount real-time rates, you can include a discount in the deal. [Figure 1–17](#) shows a deal that provides a 50% discount for the first month cycle fee.

Figure 1–17 Discount Example



7. After defining deal-specific values for the products, you create a plan and add the deal to the plan. In the plan, you set the credit limit. In [Figure 1–18](#), the credit limit is \$200.

Figure 1–18 Plan Attributes Example

Plan Attributes [Plan 1 - Measured Web Access with Discounts]

Name: Plan 1 - Measured Web Access with Discounts

Description:

Bill on demand

Account level deal: <No Deal>

Credit Limits

Resource	ID	Limit	Floor	Threshold
US Dollar	840	200.00	10.00	

Add... Modify... Delete

Services

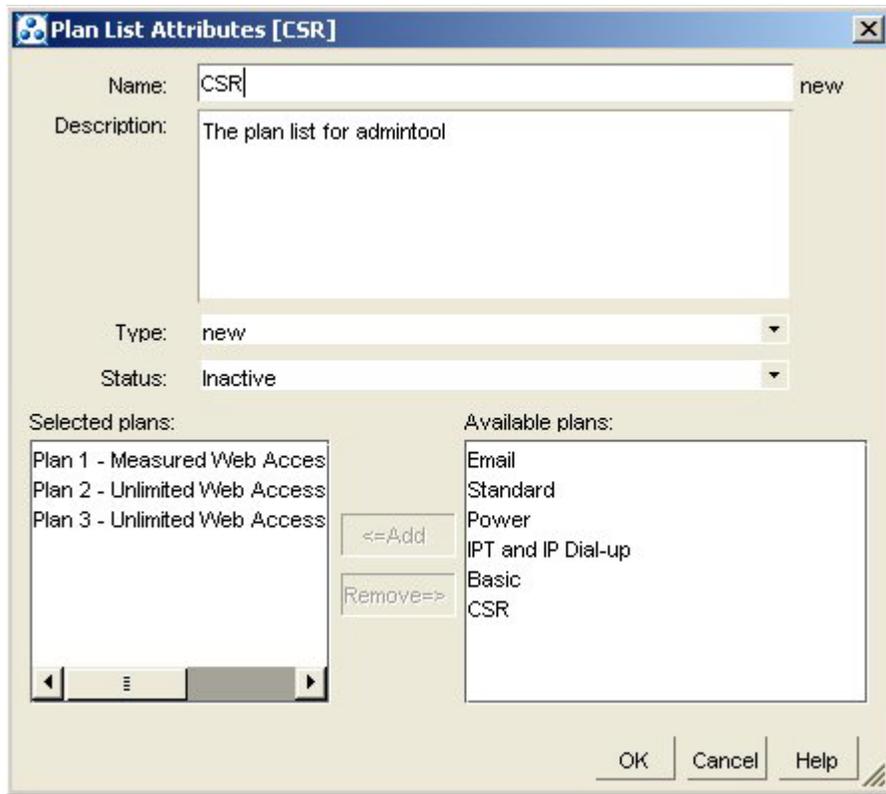
Service Type	Deal
/service/ip	Deal 1a - Measured Internet Service

Add... Modify... Delete

OK Cancel Help

8. After creating the plan, you add it to one or more plan lists. In [Figure 1–19](#), the plan is added to the **CSR - new** plan list. This plan list is displayed in Customer Center.

Figure 1–19 Plan List Attributes Example



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

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

Making Changes to Your Price List

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

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

Deleting Products

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

Logging Changes to Price Lists

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

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

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

To log price list changes:

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


```
- fm_rate log_refresh_product    1
```
3. Save the file.
4. Stop and restart the CM.

Ensuring Price List Consistency

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

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

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

Troubleshooting a Price List

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

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

Important: Pricing Center does not include or validate pipeline rating data when you save a price list.

Displaying Price List Elements

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

Note: Offer profiles cannot be displayed using Pricing Center.

Creating a Price List with the XML Pricing Interface

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

Common Price List Solutions

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

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

Note: These topics apply to real-time rating only.

Providing Free Hours

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

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

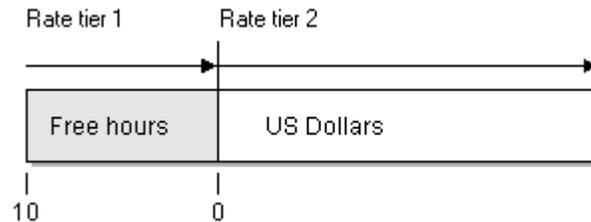
For example:

- Cycle forward rate plan: Credit 10 hours per month
- Balance impact: -10 hours every month
- Usage rate plan:
 - Rate tier 1: Charge hours. This tier is valid until the hours in the customer's account reaches 0.
 - Balance impact: 1 hour for every hour online
 - Rate tier 2: Charge dollars. This tier is valid when rate tier 1 can no longer be applied.
 - Balance impact: 1 dollar for every hour online.

- Fold rate plan: Delete unused hours (see "Deleting Unused Free Hours").

Figure 1–20 shows how the free hours are used first:

Figure 1–20 Free Hours Example



Deleting Unused Free Hours

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

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

Figure 1–21 shows the balance impact in Pricing Center. The resource is hours and the balance impact is -1, which removes one hour for every hour in the account balance.

Figure 1–21 Balance Impact that Deletes Unused Hours

	Resource ID	GLID	Impact Category	S	P	D	Fixed Amount	Scaled Amount	Units
1	async bulk hours [100...	undefine...		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	0.00	-1.00	None [0]

Charging for Canceling a Product

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

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

Charging a Discounted Purchase Fee Based on Date of Purchase

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

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

For example:

- Rate tier 1: Discounted purchase fee
- Priority: high

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

Creating a “Third-Month Free” Rate

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

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

For example:

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

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

The relative duration end date is 65 days after purchase to ensure that the rate tier is not valid in the fourth billing cycle.

- Rate tier 2: Normal cycle fee
- Balance impact: 19.95 US dollars

Creating Discounts Based on Usage

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

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

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

- Quantity Discount Bracket 1: Balance impacts for first 100 email messages
- Minimum quantity: none
- Maximum quantity: 100
- Balance impact resource: US dollars
- Scaled impact: .02 US dollars
- Quantity Discount Bracket 2: Balance impacts for email messages after the first 100
- Minimum quantity: 100
- Maximum quantity: none

- Balance impact resource: US dollars
- Scaled impact: .01 US dollars

Creating Products for Administrative Events

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

When you create the products:

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

Advanced Rating Features

For information about advanced rating features, see:

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

Note: These topics apply to real-time rating only.

Charging Cycle Forward Fees in Advance

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

When you define your event map in Pricing Center, you can specify how far in advance to bill in days, weeks, or months, as shown in [Figure 1-22](#):

Figure 1-22 *Advance Billing Example*

The screenshot shows a configuration window titled "In Advance Billing". It contains two radio button options. The first option is "Don't bill in advance" and is unselected. The second option is "Charge cycle fees" and is selected. To the right of the "Charge cycle fees" option, there is a text input field containing the number "14", a dropdown menu set to "Days", and the text "in advance of billing cycle".

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

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

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

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

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

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

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

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

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

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

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

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

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

About Charging for Administrative Events

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

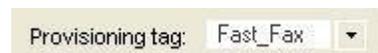
About Using Product-Level Provisioning to Configure Services

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

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

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

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



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

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

To create provisioning tags, see "[Working with Provisioning Tags](#)".

About Remittance

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

You must create remittance products to use the remittance feature. For more information, see "Remitting funds to third parties" in *BRM Configuring and Running Billing*.

About Custom Event Analysis

Custom event analysis enables you to perform rating based on:

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

Setting up custom event analysis requires modifying or creating a policy opcode and editing and loading several configuration files. For more information, see "[About Charging for Custom Events and Attributes](#)".

Setting Optional Rating Flags

You can turn a number of optional rating features on and off by changing entries in the Connection Manager (CM) configuration file (*BRM_Home/sys/cm/pin.conf*):

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

To use these features:

1. Open the Connection Manager (CM) **pin.conf** file in *BRM_Home/sys/cm*.
2. Change the value of the **extra_rate_flags** entry.

Each feature has a different hexadecimal value shown in [Table 1-9](#):

Table 1-9 Optional Rating Features and Values

Optional Rating Feature	Value	Effect
Fixed discount option	0x01	If present, discounts are calculated using this formula: $[(rate - discount) * quantity]$ If absent, discounts are calculated using this formula: $[(rate * quantity) - discount]$

Table 1–9 (Cont.) Optional Rating Features and Values

Optional Rating Feature	Value	Effect
Credit floor checking	0x02	If present, credit floor checking is disabled. If absent, credit floor check is enabled.
Return extend information	0x10	If present, extended information is returned with the rating result. If absent, extended information is not returned.

To use more than one option, add the values for each option and use the sum value.

You do not need to restart the CM to enable this entry.

Understanding the Sample Pricing Plans

This chapter describes the sample pricing plans included with the Oracle Communications Billing and Revenue Management (BRM) Pricing Center. For information about creating a price list, see "[About Creating a Price List](#)".

About the Sample Plans

The following sections describe the sample price plans.

Purpose of the Sample Plans

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

Looking at the Sample Plans

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

Note: Offer profile data cannot be accessed using Pricing Center. Use the `loadpricelist` utility or the `PCM_OP_OFFER_PROFILE_SET_OFFER_PROFILE` opcode.

- Basic BRM plans for Internet service providers (**basicISP.ipl**) are located by default in the **Sample_Price_Plans** subdirectory of the Pricing Center installation directory. The default installation directory is **C:\Program Files\Portal Software\PricingCenter\Sample_Price_Plans**.
- Plans that are based on a BRM optional manager are located by default in the **Optional_Manager_Plans** subdirectory, for example, **C:\Program Files\Portal Software\PricingCenter\Sample_Price_Plans\Optional_Manager_Plans**.

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

- The advanced sample plans are located by default in the **Sample_Price_Plans** subdirectory of the Pricing Center installation directory. The default installation directory is **C:\Program Files\Portal Software\PricingCenter\Sample_Price_Plans**.

Opening an IPL Price List File

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

Opening an XML Price List File

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

You must connect to a database to access configuration information, such as resources, G/L IDs, and valid events for services.

4. Enter the database information in the Welcome dialog box.

Tip: Click the **Help** button for information on login values.

Descriptions of the Sample Plans

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

For step-by-step procedures on creating these plans, see "[Re-Creating the Sample Plans](#)".

Plans for Internet Service Providers

This section describes the sample plans for internet service providers.

Plan 1 – Measured Web Access with Discounts

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

Target customer

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

Concepts illustrated

- Simple monthly (cycle forward) fee for Internet access
- Prorating of cycle fees
- Measured rates
- Time-of-day usage rates
- Priority assignments for multiple rates

- Product purchase limited to a set maximum

Key elements of plan

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

To re-create this plan

See "[Re-Creating Plan 1 – Measured Web Access with Discounts](#)".

Plan 2 – Unlimited Web Access with Discounts

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

Target customer

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

Concepts illustrated

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

Key elements of plan

- Internet access service
- Sign-up fee of \$15.00
- Sign-up fee discounted by 50% through May, 2000
- \$19.95 monthly rate, billed on the first day of the month
- Third month free
- One email address included

To re-create this plan

See "[Re-Creating Plan 2 – Unlimited Web Access with Discounts](#)".

Plan 3 – Unlimited Internet Service with Recurring Discounts

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

Target customer

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

Concepts illustrated

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

Key elements of plan

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

To re-create this plan

See "[Re-Creating Plan 3 – Unlimited Internet Service with Recurring Discounts](#)".

Re-Creating the Sample Plans

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

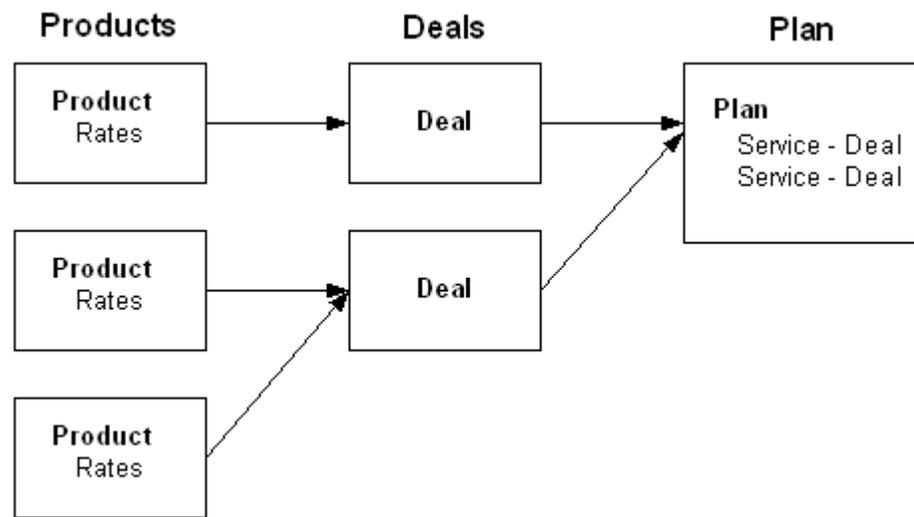
Overview of Creating a Pricing Plan

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

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

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

[Figure 2–1](#) shows the building blocks of a pricing plan:

Figure 2–1 Pricing Plan Building Blocks

To create a plan, follow these general steps:

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

The descriptions of the sample plans show the key elements of these plans.

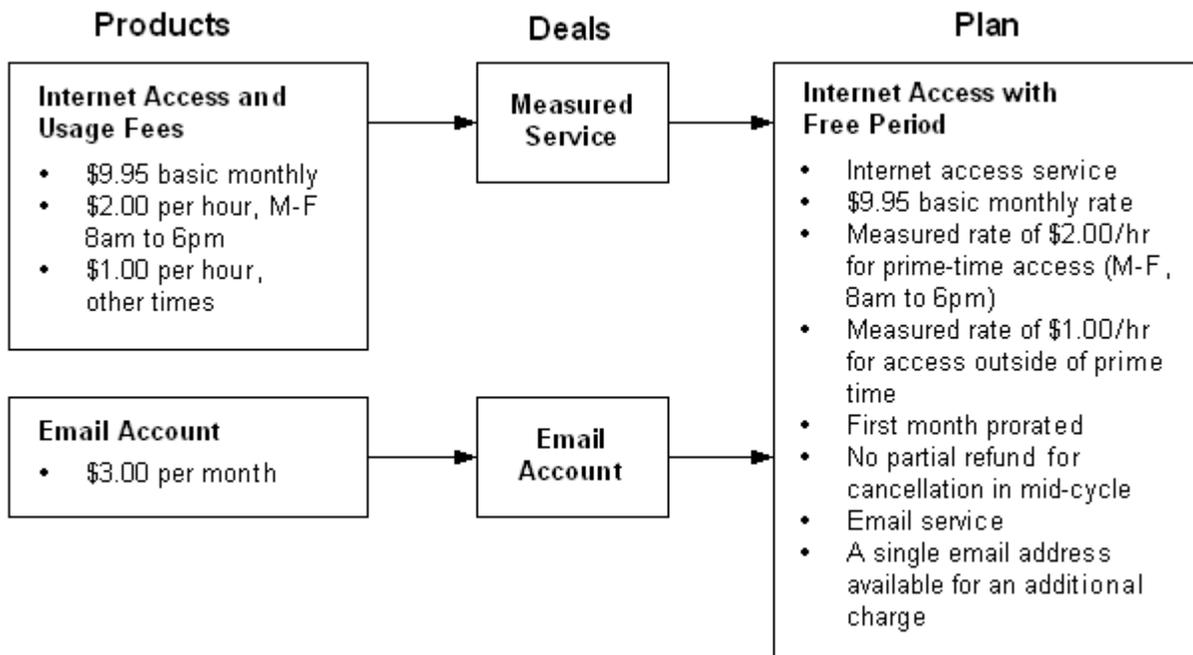
2. Use Pricing Center to create a product for each key element of your plan. Usually, each service you offer uses one product to contain all rates for that service.
3. Use Pricing Center to group the products into one or more deals.
4. Use Pricing Center to create a pricing plan that associates the deals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

Plans for Internet Service Providers

[Figure 2–2](#) show the Measured Web Access with Discounts Plan used in the following example.

Re-Creating Plan 1 – Measured Web Access with Discounts

Figure 2–2 Measured Web Access with Discounts Plan



Putting together Plan 1

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

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

Product 1a – Internet Access

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

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

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

Product 1a - Defining the Product

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

Table 2–1 Defining the Product

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

Product 1a - Creating the Monthly Internet Fee

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

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

Table 2–2 Creating the Monthly Internet Fee

Do This	Why
In the Event Map area of the Product Attributes window, click Add . In the Event column, select Monthly Cycle Forward Event .	To create the monthly access fee by associating the rate details with the cycle event, as specified by the plan. When you select Monthly Cycle Forward Event , Pricing Center automatically selects Occurrence in the Measured By column and Single Rate Plan in the Rate Plan Structure column. Leave these values as they are.
Click Open Rate Plan .	To set up the rates for this product.
In the Plan Detail tab, enter cycle_forward_monthly in the Name field.	The rate plan name identifies the rate plan in the product and reminds you of its purpose.

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

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

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

Do This	Why
In the Scaled Amount column, type 9.95 .	Recurring charges, such as the monthly fee in this product, are called <i>scaled</i> rates. Enter the dollar amount of the charge, which is \$9.95 per month.
In Units , select None [0] .	This rate uses the Occurrence ratable usage metric (RUM), which is not measured in units.
Click OK .	You are finished creating the monthly fee and now must create the usage fees.

Product 1a - Creating the Measured Usage Fees

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

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

Table 2–3 Creating the Measured Usage Fees

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

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

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

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

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

Product 1b – Email Account

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

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

Product 1b - Defining the Product

Table 2–4 Defining the Product

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

Product 1b - Creating the Monthly Email Fee

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

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

Table 2–5 Creating the Monthly Email Fee

Do This	Why
In the Event Map group, click Add . In the Event Type column, select Monthly Cycle Forward Event .	To create the monthly access fee by associating the rate details with the cycle event, as specified by the plan. When you select Monthly Cycle Forward Event , by default Pricing Center selects Occurrence in the Measured By column. You cannot change this value. Note: The values in the Rating Details group are only used for time-dependent fees.
In the Rate Plan Structure column, select Single Rate Plan . Under Single Rate Plan , click Open Rate Plan .	To set up the rates for this product.
In the Plan Detail tab, enter cycle_forward_monthly in the Name field.	The rate plan name identifies the rate plan in the product and reminds you of its purpose.
In the Rate Structure tree view, select Tier 1 . In the Tier Detail tab, enter a name for the rate tier: Monthly Email Fee .	The name identifies the rate tier in the product and reminds you when its valid.
Under Tier Detail , select Absolute or Relative , and then Starts Immediately and Never Ends . Also, under Use Time/Day Restrictions , select No restrictions .	To make the rate available indefinitely, so it never expires in the BRM database and the fee is always charged.

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

Do This	Why
In the Rate Structure tree view , select Rate 1 . In the Rate Detail tab , enter a name for the rate: \$3 a month . Note: Rate names must be unique. In the /rate storable class, the rate name is specified in the PIN_FLD_DESCR field.	To remind you of the purpose of this rate.
Select Overrides credit limit .	To continue to charge for the email account when customers exceed the credit limit.
In the Based on list, select Continuous .	This option is the default when you use only one quantity discount bracket to rate an event.
Click the Proration tab.	To specify how to prorate the charges.
In the Purchase Proration Information group, click Calculate the charge based on the amount used .	To charge customers only for the portion of the first month when they owned the product. Note: Most companies base the monthly accounting cycle on the day the customer purchases the product rather than on a calendar month, so proration is not necessary.
In the Cancel Proration Information group, click Charge for the entire cycle .	To specify that the customer is charged the full monthly fee, even if the customer cancels the service in the middle of the accounting cycle.
In the Rate Structure tree view under \$3 a month , select No Minimum - No Maximum to display the Balance Impacts tab.	This displays the Balance Impacts tab, where you define the fee.
Under Valid Quantities , select None for both the Minimum and Maximum fields.	The monthly fee is not based on a specific amount of the event.
Under Balance Impacts , click Add . Tip: Drag the right edge of any column heading to make the list wider.	To add a row to the impacts list and to specify the balance impacts - what you charge the customer for the service.
Click the Resource ID box and scroll to select US Dollar [840] .	To charge in US dollars.
In the GLID list, select Monthly Fees [102] .	To record the charges as monthly fees in your general ledger.
Click S for Sponsorable .	To enable the rate to be paid by another account. Select this option to enable one account to pay for another.
In the Scaled box, type 3 .	Recurring fees, such as the monthly fee in this product, are called <i>scaled</i> rates. You enter the dollar amount of the fee.
In Units , select None [0] .	This rate uses the Occurrence ratable usage metric (RUM), which is not measured in units.
Click OK until you return to the Document window.	You are finished creating the email product.

Deal 1a – Measured Internet Service

1. In Pricing Center, choose **Insert – Deal**.
2. Name the deal: **Deal 1a – Measured Internet Service**. (You can also add a description.)

3. In **Applies to**, select **/service/ip**.
4. In the **Products Associated with Deal** group, click **Add**.
5. In the **Product Name** list, select **Product 1a – Internet Access**.
6. Click **OK**.

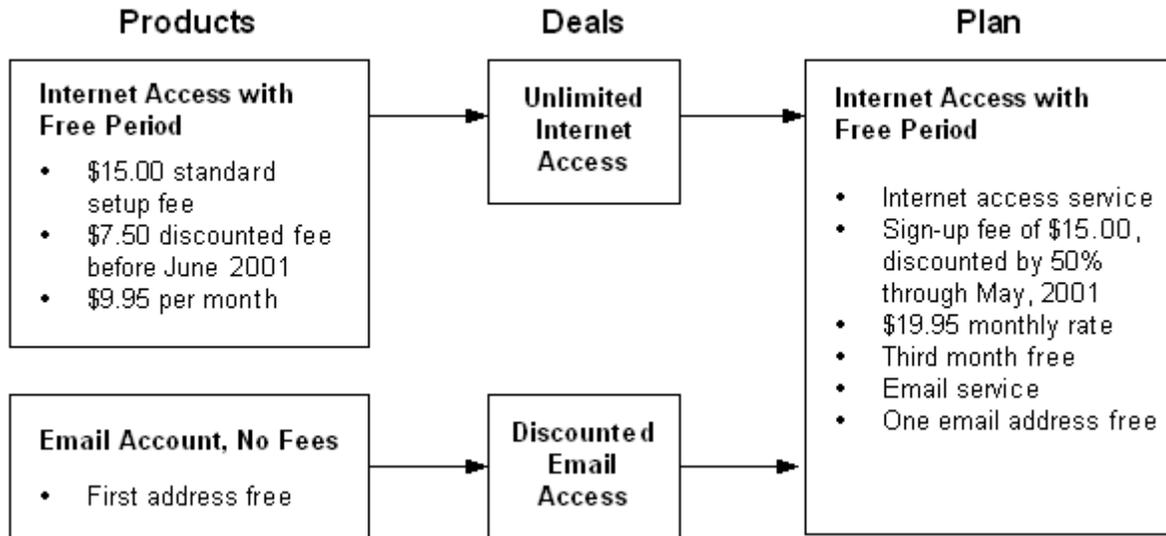
Deal 1b – Standard Email Access

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

Re-Creating Plan 2 – Unlimited Web Access with Discounts

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

Figure 2–3 Unlimited Web Access with Discounts Plan



Putting together Plan 2

1. Create a product to charge the sign-up fee and monthly Internet access fee. See "[Product 2a – Internet Access with Free Period](#)".
2. Create a product for the email account. See "[Product 2b – Email Account, No Fees](#)".
3. Create a deal for the Internet access. See "[Deal 2a – Unlimited Internet Service](#)".
4. Create a deal for the email account. See "[Deal 2b – Discounted Email Access](#)".
5. In Pricing Center, choose **Insert – Plan**.
6. Name the plan: **Plan 2 – Unlimited Web Access with Discounts**. (You can also add a description.)

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

Product 2a – Internet Access with Free Period

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

Product 2a - Defining the Product

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

Table 2–6 Defining the Product

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

Product 2a - Creating the Sign-up Fees

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

Product 2a - Creating the Discounted Sign-up Fee

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

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

Table 2-7 Defining the Product

Do This	Why
In the Event Map group, click Add . In the Event column, select Product Purchase Fee Event .	To create the sign-up fee by associating the rate details with the purchase event, as specified by the plan. When you select Product Purchase Fee Event , Pricing Center automatically selects Occurrence in the Measured By column. You cannot change this value. Note: The values in the Rating Details group are used only for time-dependent fees.
In the Rate Plan Structure column, select Single Rate Plan . Under Single Rate Plan , and click Open Rate Plan .	To set up the rates for this product.
In the Plan Detail tab, enter purchase in the Name field.	The rate plan name identifies the rate plan in the product and reminds you of its purpose. For system events, you must enter the exact name of the rate plan.
In the Rate Structure tree view, select Tier 1 . In the Tier Detail tab, enter a name for the rate tier: Discounted IP Sign-up Fee .	The name identifies the rate tier in the product and reminds you that it is a discounted fee.
Under Effective Dates , select Absolute .	To make the rate available for a limited time, and to ensure that it expires on a certain date.
Under Ends , select At , then On , and set the end date to 03/01/2001 .	The discount is valid through the end of February, 2001. Valid date ranges do not include the end date; therefore, you enter the first date when this rate is no longer valid.
Under Use Time/Day Restrictions , select No restrictions .	To bypass setting valid days of the week.
In the Rate Structure tree view, select Rate 1 . In the Rate Data tab, enter a rate name: \$7.50 before 3/1/01 . Note: Rate names must be unique. In the /rate storable class, the rate name is specified in the PIN_FLD_DESCR field.	To remind you of the purpose of this rate.
Select Overrides credit limit .	To continue to charge for Internet access when customers exceed the credit limit.
In the Based on list, select Continuous .	This option is the default when you use only one quantity discount bracket to rate an event.
In the Rate Structure tree view, select No Minimum -No Maximum .	This displays the Balance Impacts tab, where you define the cost of the rate.
Under Valid Quantities , select None for both the Minimum and Maximum fields.	The monthly fee is not based on a specific amount of the event.
Under Balance Impacts , click Add . Tip: Drag the right edge of any column heading to make the list wider.	To add a row to the impacts list and to specify the balance impacts - what you charge the customer for the service.
Click the Resource ID row and scroll to select US Dollar [840] .	To charge in US dollars.
Click the GLID row and scroll to select Purchase Fees [101] .	To record the fees as purchase fees in your general ledger.

Table 2–7 (Cont.) Defining the Product

Do This	Why
Select S for Sponsorable .	To enable the fee to be paid by another account. Select this option to enable one account to pay for another.
In the Fixed Amount field, type 7.5.	One-time charges, such as the purchase fee in this product, are called <i>fixed</i> rates. You enter the dollar amount of the charge.
In the Rate Structure tree view, select Purchase .	You are finished specifying this rate tier.

Product 2a - Creating the Standard Sign-up Fee

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

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

Table 2–8 Creating the Standard Sign-up Fee

Do This	Why
In the Rate Structure tree view, select Purchase and then click Add .	To add a rate tier to the rate plan.
In the Rate Structure tree view, select New Rate Tier . In the Tier Detail tab, enter a name for the rate tier: Standard .	The name identifies the rate tier in the product and reminds you of the purpose of this rate.
Under Tier Detail , select Absolute or Relative , then Starts Immediately and Never Ends . Also, under Use Time/Day Restrictions , select No restrictions .	To make the rate available indefinitely, so it never expires in the BRM database.
In the Rate Structure tree view, select New Rate . In the Rate Data tab, enter a name for the rate: \$15 Sign-up Fee . Note: Rate names must be unique. In the /rate storable class, the rate name is specified in the PIN_FLD_DESCR field.	To remind you of the purpose of this rate.
Select Overrides credit limit .	To continue to charge for the email account if customers exceed the credit limit.
In the Based on list, select Continuous .	This option is the default when you use only one quantity discount bracket to rate an event.
In the Rate Structure tree view, select No Minimum - No Maximum .	This displays the Balance Impacts tab, where you define the cost of the purchase rate.
Under Valid Quantities , select None for both the Minimum and Maximum fields.	The monthly fee is not based on a specific amount of the event.
Under Balance Impacts , click Add .	To add a row to the impacts list and to specify the balance impacts - what you charge the customer for the service.
Click the Resource ID box and scroll to select US Dollar [840] .	To charge in US dollars.
In the GLID list, select Purchase Fees [101] .	To record the charges as one-time purchase fees in your general ledger.

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

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

Product 2a - Creating the Cycle Fees

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

Product 2a - Creating the Free Month

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

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

Table 2–9 Creating the Free Month

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

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

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

Product 2a - Creating the Standard Monthly Fee

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

[Table 2–10](#) describes the steps for creating the standard monthly fee.

Table 2–10 Creating the Standard Monthly Fee

Do This	Why
In the Rate Structure tree view , select the cycle_forward_monthly rate plan and click Add .	To add a new rate tier to the existing cycle_forward_monthly rate plan.

Table 2–10 (Cont.) Creating the Standard Monthly Fee

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

Product 2b – Email Account, No Fees

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

Product 2b - Creating the Product

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

Table 2–11 Creating the Product

Do This	Why
In Pricing Center, choose Insert – Product and select Create a new product .	To create a product.
Click Next and give the product a name: Product 2b – Email Account, No Fees . (You can also add a description.)	The product name identifies the product in Pricing Center. This name must be unique to all products in the same database brand.
When asked what type of product to create, select Subscription and in Applies To , choose /service/email .	This enables the product to contain recurring monthly rates and associates it with the email service.

Table 2–11 (Cont.) Creating the Product

Do This	Why
Continue through the product creation wizard until you get to the Valid Purchase Quantity group.	Although you can change various attributes of the product while using the wizard, this sample product uses the default attributes up to this point.
Click Maximum of and enter 1 .	To specify that a customer can purchase no more than one email account.
Finish the product creation wizard, and click Yes at the end.	After you finish the wizard, you specify the rates for this product.

Product 2b - Creating the Free Rate

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

[Table 2–12](#) describes the steps for creating the free rate.

Table 2–12 Creating the Free Rate

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

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

Do This	Why
In the Rate Structure tree view, select No Minimum- No Maximum to display the Balance Impacts tab.	This displays the Balance Impacts tab, where you define the cost of the rate.
Under Valid Quantities , select None for both the Minimum and Maximum fields.	The monthly fee is not based on a specific amount of the event.
Under Balance Impacts , click Add . Tip: Drag the right edge of any column heading to make the list wider.	To add a row to the impacts list and to specify the balance impacts - what you charge the customer for the service.
Click the Resource ID box and scroll to select US Dollar [840] .	To bill in US dollars.
In the GLID list, select Monthly Fees [102] .	To record the charges as monthly fees in your general ledger.
For Fixed and Scaled , make sure the values are 0 .	You do not charge the customer for the third month.
In Units , select None [0] .	This rate uses the Occurrence ratable usage metric (RUM), which is not measured in units.
Click OK .	You are finished creating the email product.

Deal 2a – Unlimited Internet Service

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

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

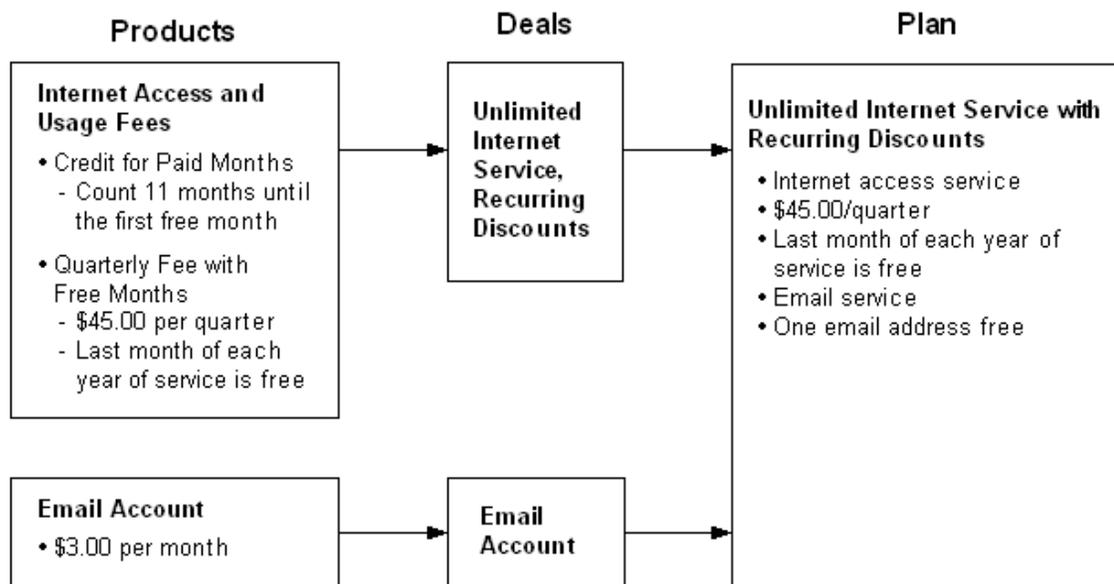
Deal 2b – Discounted Email Access

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

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

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

Figure 2–4 Unlimited Internet Service with Recurring Discounts Plan



Putting together Plan 3

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

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

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

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

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

As with products, you also can use a deal from another plan as a building block in this plan.

13. Click **OK**.
14. In the **Credit Limits** section, click **Add**.
15. In the **Resource ID** list, select **Paid Quarter**.
16. In the **Credit Limit** field, de-select **None**, then type **3**.
17. Click **OK**.
18. Click **OK** again to finish the plan.

Creating a custom resource for Plan 3

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

Table 2–13 Creating a Custom Resource

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

Product 3a – Prepaid Internet Service with Free Period

This product serves several purposes:

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

Product 3a - Defining the Product

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

Table 2–14 Defining Prepaid Internet Service with Free Period

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

Product 3a – Creating the Quarterly Fee

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

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

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

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

Table 2–15 Determining Fee

Do This	Why
In the Event Map group, click Add . In the Event column, select Quarterly Cycle Forward Event .	To specify billing at the beginning of each three-month cycle, as specified by the plan.
In the Measured By column, select Occurrence . See ratable usage metric (RUM).	To specify that the rate is charged every three months when the quarterly cycle forward event occurs. Note: The rating details are used only for time-dependent fees.
In the Rate Plan Structure column, select Single Rate Plan . Under Single Rate Plan , click Open Rate Plan .	To set up the rates for this product.
In the Plan Detail tab, enter cycle_forward_quarterly in the Name field.	The rate plan name identifies the rate plan in the product and reminds you of its purpose. For system events, you must enter the predefined name of the rate plan.

Deal 3 – Unlimited Internet Service, Recurring Discounts

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

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

Descriptions of the Advanced Sample Plans

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

ASP User Profile Plan

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

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

Concepts Illustrated

Real-time volume discounting and minimum charge.

Customization Required for the ASP User Profile Plan

The ASP User Profile plan includes the following custom resources:

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

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

Adding custom resources

To add the custom resources:

1. Start the Configuration Center and connect to your BRM database.
2. Ensure that your database does not include custom resources with any of the resource IDs above: 1000022 through 1000025.

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

3. Add the resources.

Structure of the ASP User Profile Plan

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

add_active_user rate plan

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

add_casual_user rate plan

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

add_selfservice_user rate plan

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

del_active_user rate plan

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

del_casual_user rate plan

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

Fold rate plan selector

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

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

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

- The fifth quantity discount bracket applies if the balance of Pre-Discount Dollars is greater than 50,000. It multiplies the Pre-Discount Dollar resource balance by 0.8 and folds it into the US Dollars resource balance. This is, in effect, a 20% discount for spending more than 50,000 Pre-Discount Dollars.

Questions about the Plan

Why is the Pre-Discount Dollars custom resource necessary?

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

Audio/Video Plan

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

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

Concepts Illustrated

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

Customization Required for the Audio/Video Plan

The Audio/Video plan includes two custom resources:

- Minute Streamed Audio (Resource ID = 1000005)
- Minute Streamed Video (Resource ID = 1000006)

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

Adding custom resources

To add the custom resources:

1. Start the Resource Editor and connect to your BRM database.
2. Ensure that your database does not already include custom resources with any of these Resource IDs: 1000005 or 1000006.

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

3. Add the resources.

Structure of the Audio/Video Plan

The plan contains three products: AudioVideo Monthly, AudioVideo Audio Usage, and AudioVideo Video Usage.

AudioVideo Monthly product

This product contains four rate plans. Two are single rate plans and two are combined in a rate plan selector.

The Monthly Cycle Forward Event rate plan contains an AudioVideo Monthly Rate Tier that charges \$14.95 per month for the plan.

The Product Purchase Fee Event rate plan contains a Purchase Rate Tier that discounts the first month \$14.95 when the plan is purchased. In effect, this makes the first month of service free.

The Cycle Fold Event rate plan selector combines two rate plans:

- The first rate plan, called Audio Fold, contains an Audio Fold Rate Tier that multiplies the balance of the Minute Streamed Audio resource by negative one (-1) and adds it to the resource balance. This resets the Minute Streamed Audio resource balance to zero.
- The second rate plan, called Video Fold, contains a Video Fold Rate Tier that multiplies the balance of the Minute Streamed Video resource by negative one (-1) and adds it to the resource balance. This resets the Minute Streamed Video resource balance to zero.

AudioVideo Audio Usage

This product contains one rate plan, IP Audio Event, with two rate tiers, Audio Basic Minutes and Audio Excess Minutes:

- The Audio Basic Minutes tier charges nothing for each minute of audio streamed up to 120 minutes per month.
- The Audio Excess Minutes tier charges a variable rate for each minute of audio streamed over 120 minutes per month. The rate structure includes two time of day periods: Peak and Off Peak. During Peak hours, 8:30 a.m. to 6:00 p.m., streamed audio is charged at 25 cents a minute. During Off Peak hours, 6:00 p.m. to 8:30 a.m., streamed audio is charged at 15 cents a minute.

AudioVideo Video Usage

This product contains one rate plan, IP Video Event, with two rate tiers, Video Basic Minutes and Video Excess Minutes:

- The Video Basic Minutes tier charges nothing for each minute of video streamed up to 60 minutes per month.
- The Video Excess Minutes tier charges a variable rate for each minute of video streamed over 60 minutes per month. The rate structure includes two time-of-day periods: Peak and Off Peak. During Peak hours, 8:30 a.m. to 6:00 p.m., streamed video is charged at 35 cents a minute. During Off Peak hours, 6:00 p.m. to 8:30 a.m., streamed video is charged at 25 cents a minute.

Questions about the Plan

Why not just delay the initial monthly charge for one month instead of including a purchase product that refunds the first month's charge?

By including a purchase product that refunds the first month's charge, you can quickly change the plan to offer only a partial refund in the first month.

Customer Service Plan

This plan (**CustomerService.IPL**) is designed to let one company offer hosted customer service applications to a second company. The second company is then charged a fee for each subscriber to the service and is also charged for customer service minutes used by their subscribers.

The second company is charged a fee per subscriber and is also given a certain amount of customer service minutes per subscriber based on the total number of subscribers the second company has. The number of subscribers is checked when the Customer Service plan is first purchased and each time BRM billing is run - typically at the end of each month.

The charge per subscriber and the number of free customer service minutes given is determined as shown in [Table 2-16](#):

Table 2-16 Subscribers and Charges

Number of Subscribers	Charge per Subscriber	Number of Free Customer Service Minutes per Subscriber
0 to 5000	\$5	2
5000 to 10000	\$4	3
10,000+	\$3	4

Concepts Illustrated

Tiered pricing, products supporting business to business scenarios.

Customization Required for the Customer Service Plan

The Customer Service plan includes one custom service:

- **/service/customerservice**

The Customer Service plan includes two different custom resources:

- Customer Service Minute (resource ID = 1000026)
- Subscription (resource ID = 1000502)

The Customer Service plan also includes two custom events:

- **event/activity/add_subscription**
- **event/activity/delete_subscription**

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

Adding custom services

You must use the Developer Workshop to add the **/service/customerservice** custom service to your BRM database.

Adding custom resources

To add the custom resources:

1. Start the Resource Editor and connect to your BRM database.
2. Ensure that your database does not include custom resources with any of the resource IDs above: 1000026 or 1000502.

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

3. Add the resources.

Adding custom events

This plan includes two custom events: **event/activity/add_subscription** and **event/activity/delete_subscription**.

You must use the Developer Workshop and edit **pin_event_map** to add these custom events to your BRM database.

Structure of the Customer Service Plan

The plan contains one product, Customer Service Product, with four rate plans, Add Subscription Event, Delete Subscription Event, Monthly Cycle Forward Event, Session Event, and one rate plan selector, Cycle Fold Event.

Add Subscription Event

This rate plan adds one to the Subscription resource when an **event/activity/add_subscription** event occurs.

Delete Subscription Event

This rate plan deletes one from the Subscription resource when an **event/activity/delete_resource** event occurs.

Monthly Cycle Forward Event

This rate plan has a customer service rate tier that both charges per subscription and allots free minutes based on the number of total subscriptions as shown in [Table 2-17](#):

Table 2-17 Subscribers and Charges for Monthly Cycle Forward Event

Number of Subscribers	Charge per Subscriber	Number of Free Customer Service Minutes per Subscriber
0 to 5000	\$5	2 (negative 2 added to the resource balance)
5000 to 10000	\$4	3 (negative 3 added to the resource balance)
10,000+	\$3	4 (negative 4 added to the resource balance)

Session Event

This rate plan has one rate tier, the Customer Service Usage Rate Tier, that adds one to the Customer Service Minute resource balance for each customer service minute used by the company’s subscribers.

Cycle Fold Event

This rate plan selector has one rate plan, Fold Minutes. The rate plan folds the Customer Service Minute resource balance to US Dollars, \$1 for each 1 minute used over the minutes given by the Cycle Forward Monthly Event rate plan. Then it folds the Customer Service Minute resource balance to zero by multiplying the balance by negative 1 and then adding it to the balance.

Note that the order in which the folds occur is very important. If the Customer Service Minute resource balance is folded first before the number of US Dollars is calculated, the company will never be charged.

Questions about the Plan

What happens if the number of subscribers during a month increases enough to qualify for a lower rate per subscriber and more customer service minutes per subscriber?

The rate tier is only checked when the plan is first purchased and each month when billing is run. For example, if a company starts a month with 4500 subscribers and then adds 700 subscribers during the month, they will still be charged per subscriber and given customer service minutes per subscriber based on having 4500 subscribers until the company runs billing again.

If all of the customer service minutes given to a company for the number of subscribers they have are not used by the end of the month, why does not the fold cause a credit to be issued to US Dollars?

The fold quantity bracket starts at "0" with no maximum so it is not activated if the customer service minute balance is negative. If you actually wanted a credit to be issued for customer service minutes not used, you could add another bracket to the fold that starts at no minimum and ends at "0".

IP Limited Access Plan

This plan (**IPLimitedAccess.IPL**) gives 100 hours of IP access for \$19.95 per month. Each hour of IP access after a 100 hours is charged at \$2.00 an hour. The first 12th month of IP service is free. Unused hours in each month are lost.

For every IP access hour used, the customer gets 1 frequent flier credit at the end of the billing period. In addition, for every IP access hour in excess of 100 in a month, the customer is given 1 game credit which must be used by the end of that month.

The plan also includes unlimited email service for a one-time purchase fee of \$5.00.

Concepts Illustrated

Free hours, free 12th month, hourly credits, folds

Customization Required for the IP Limited Access Plan

The IP Limited Access plan includes several different custom resources:

- Bulk hours (Resource ID = 1000013)
- paid month (Resource ID = 1000008)
- Frequent Flier Miles (Resource ID = 1000003)
- Free Game (Resource ID = 1000004)

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

Adding custom resources

To add the custom resources:

1. Start the Resource Editor and connect to your BRM database.
2. Ensure that your database does not include custom resources with any of the Resource IDs above: 1000003, 1000004, 1000008, or 1000013.

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

3. Add the resources.

Structure of the IP Limited Access Plan

The plan contains three products: IP Limited Access Email, IP Limited Access Monthly, and IP Limited Access Usage.

IP Limited Access Email product

This is an item product consisting of a one-time charge of \$5.00 for unlimited email service.

IP Limited Access Monthly product

This is a subscription product containing two event types: Monthly Cycle Forward Event and Cycle Fold Event.

Monthly Cycle Forward Event

This includes a monthly rate tier with three quantity discount brackets.

- The first bracket charges \$19.95 per month and increments the Paid Month resource by 1 each month.
- When Paid Months reaches 11, the second quantity discount bracket takes effect. The second bracket increments the Paid Month Resource by 1 but does not charge for that month. This bracket is valid only when Paid Month is equal to or greater than 11 and less than 12, so it gives the customer the 12th month of service free.
- The third quantity bracket takes effect after the 12th month of service. Since this bracket is valid when Paid Month is equal to or greater than 12, it remains in effect indefinitely after the 12th month of service. It affects resources the same way the first bracket does, thereby ensuring that only the first 12th month rather than every 12th month of service is free.

Cycle Fold Event

The Cycle Fold Event includes two fold rates which zero out the game credit balance at the end of the month and any hours of IP usage accrued in the month.

The folds zero out any remaining game credits and hours of IP service by multiplying the appropriate resource balance by -1 and adding it to the resource balance. For example, if a customer used 35 Bulk Hours of IP service and has 3 game credits left at the end of a month, the respective resource balances are 35 and 3. The fold rates multiply those remaining resource values by -1, resulting in -35 and -3 respectively, and adds those results to the resource balances to end with (35 + negative 35) or 0 Bulk hours and (3 + negative 3) or 0 Game Credits.

IP Limited Access Usage product

This is a subscription product that includes one event type: Session Event.

Session Event includes two rate tiers.

- Base Rate Tier
 - This rate tier is in effect until the Bulk Hours resource for a month reaches "100.00".

In this tier, for each hour of service, the Bulk Hours resource is incremented by 1 and the Frequent Flier miles resource is incremented by 1. Thus, each hour decreases the number of base IP hours available and adds 1 frequent flier mile to the customer's account. The hours in the Base Rate Tier, the first 100 hours of IP usage in a month, have a balance impact of "0" because there is no usage fee for the first 100 hours in a month.

- **Excess Rate Tier**

This rate tier takes effect after the first 100 hours of IP usage in a month.

In this tier, for each hour of service, the customer is charged \$2 and given 1 frequent flier mile and 1 game credit.

Questions about the Plan

How do I make every 12th month of base 100 hours of IP service free instead of just the first 12th month?

To make every 12th month free instead of just the first 12th month, change the quantity discount brackets within the Monthly Cycle Forward Event of the IP Limited Access Monthly product:

1. Leave the first quantity discount bracket as is.
2. Delete the third bracket.
3. Change the second quantity discount bracket as follows:
 - a. Change the Maximum Valid Quantities to None.
 - b. Change the Paid Month resource impact from 1.00 to (11.00).

This zeroes out the Paid Month resource so the first bracket is used until the next 12th month of service is reached.

Since the number of Bulk Hours a customer uses is irrelevant as far as fees are concerned once they use over 100 in a month, why does the Excess Rate tier of IP Limited Access Usage continue to track the number of Bulk Hours used?

If Bulk Hours were not tracked by the Excess Rate tier, CSRs and customers would have no easy way of knowing how many excess hours the customer was charged for in a month. This way, the exact number of excess hours appears in Customer Center.

If Bulk Hours were not tracked, CSRs and customers must divide the excess charges by \$2 to find the number of excess hours used in a month. But if excess hours were charged on a sliding scale by the plan, say \$2 for the first 50 excess hours, \$1.50 for the next 50 excess hours, and \$.75 thereafter, the calculation would become more difficult and time consuming to perform. This way, the number of excess hours used is immediately available.

Online Articles Plan

This plan (**On-LineArticles.IPL**) charges \$30.00 for quarterly access to an online articles service. The first five articles downloaded per month are free. The second five articles downloaded are charged at \$2.00 an article. The next ten articles downloaded are charged at \$1.50 an article. After that, all articles are charged at \$1.00 an article.

In addition, if a customer downloads 30 or more articles in a month, the entire usage charge for that month is discounted 15%.

Concepts Illustrated

Quarterly cycle fee, variable usage fees based on monthly volume, discounting of all usage fees when a particular level of usage is reached.

Customization Required for the Online Articles Plan

The Online Articles plan includes two custom resources:

- Num Articles (resource ID = 1000015)
- Usage Charge (resource ID = 1000014)

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

Adding custom resources

To add the custom resources:

1. Start the Resource Editor and connect to your BRM database.
2. Ensure that your database does not include custom resources with any of the resource IDs above: 1000014 or 1000015.

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

3. Add the resources.

Structure of the Online Articles Plan

The plan contains one product, On-Line Articles; two rate plans, Quarterly Cycle Forward Event, and Content Download Event; and one rate plan selector, Cycle Fold Event.

Quarterly Cycle Forward Event rate plan

This rate plan charges \$30.00 per quarter. Note that the rate plan name is **cycle_fold_quarterly**.

Content Download Event rate plan

This rate plan includes four quantity discount brackets:

- The first quantity discount bracket increments the Num Articles resource by 1 for each article downloaded per month until 5 articles are downloaded.
- The second quantity discount bracket increments the Num Articles resource by 1 and increments the Usage Charge resource by 2.00 for each article downloaded per month until 10 articles are downloaded.
- The third quantity discount bracket increments the Num Articles resource by 1 and increments the Usage Charge by 1.50 for each article downloaded per month until 20 articles are downloaded.
- The fourth quantity discount bracket increments the Num Articles resource by 1 and increments the Usage Charge by 1.00 for each article downloaded over 20 per month with no maximum.

Cycle Fold Event rate plan selector

This rate plan selector contains two rate plans:

- The first rate plan, Fold Usage Rate Plan, folds the Usage Charge into US Dollars based on the number of articles downloaded. If the Num Articles resource is less than 30, the Usage Charge resource balance is multiplied by 1.00 and folded directly into US Dollars. If the Num Articles resource is 30 or greater, the Usage Charge resource balance is multiplied by .85 and then folded into US Dollars. This, in effect, gives the user a 15% discount on all of their usage charges if they download 30 or more articles in a month.
- The second rate plan, Fold Articles Rate Plan, folds the Num Articles resource to zero by multiplying the balance by negative 1 and adding it to the balance.

Note that the order in which these rate plans are applied is very important. If the Fold Articles Rate Plan is applied first, customers will never get a 15% discount on their usage regardless of how many articles they download in a month.

You can control the order in which fold rate plans are applied by making sure the resource that you want to fold first either has a lower resource ID than the other resources you want to fold or is listed above the other resources you want to fold in the `pin_beid` file.

Questions about the Plan

Why is the Usage Charge custom resource necessary?

The Usage Charge custom resource is necessary because the 15% discount for 30 or more articles downloaded a month applies only to usage charges. Therefore you need a way of tracking usage fees separately from any other fees. If you simply charged US Dollars in the Content Download Event rate plan, you must either discount all customer fees, including any quarterly fees, or not discount customer fees at all.

T1Access Plan

This plan (**T1Access.IPL**) includes a custom T1 access service. It provides the service for a monthly charge of \$5,000 and an initial setup fee of \$600. In addition, if the customer cancels the service before a year after the purchase date, they are charged a \$10,000 cancellation fee.

Concepts Illustrated

Cancellation fee, custom service, relative date ranges for tiers

Customization Required for the T1 Access Plan

The T1 Access plan includes the custom service **service/T1**.

You must use the Developer Workshop to add this custom service to your BRM database.

Structure of the T1 Access Plan

The plan contains three products: T1 Purchase, T1 Monthly, and T1 Cancellation.

T1 Purchase product

This product consists of a one-time purchase fee of \$600 for starting the T1 service.

T1 Monthly product

This product consists of a cycle forward event for \$5,000 a month for the T1 service.

T1 Cancellation product

This product consists of a one time cancellation fee of \$5,000, triggered when the product is canceled. It is set up with a valid date range of 365 days. Thus, 365 days after the plan is purchased, the cancellation product and the associated cancellation fee no longer apply.

Questions about the Plan

Why is the cancellation fee a subscription instead of an item?

The cancellation event is generated only when the containing product is part of a subscription product. This is different from a purchase fee which is an item.

Where do you specify that the cancellation fee is valid for one year?

In the cancel rate tier, the rate tier duration is set as a Relative Date Range. The cancel rate is valid immediately and expires in 365 days.

This means that no matter when the T1 Access Deal is purchased, the cancellation fee will apply for 365 days after the purchase date.

Web Hosting Plan

This plan (**WebHosting.IPL**) charges customers for Web Hosting space. The customer is given 100 MB of Web Hosting space for \$40 a month. If a customer is using more than 100 MB at the end of the month, they are sent a warning email. Each month after the first month that they exceed the limit, they are also charged 10 cents for every megabyte over 100 that have at the end of the month.

Concepts Illustrated

No usage charge up to a threshold, warning email, delayed charge for excess usage, bill excess over threshold.

Customization Required for the Web Hosting Plan

The Web Hosting plan includes the custom service `/service/diskspace/`, a custom event called Used More Disk Space, and two different custom resources:

- Warning (resource ID = 1000011)
- Peak MBs (resource ID = 1000503)

You must add this custom service, this custom event, and these custom resources to your BRM database for the plan to work.

Adding custom services

To add the `/service/diskspace/` custom service to your BRM database, use the Developer Workshop.

Adding custom events

To add the custom event Used More Disk Space, you must configure the DM, set the proper flag in Developer Workshop, map the event to the `/service/diskspace` service.

Note that the Used More Disk Space event must include custom code to set the Peak MBs resource correctly.

Adding custom resources

To add the custom resources:

1. Start the Resource Editor and connect to your BRM database.

2. Ensure that your database does not include custom resources with the resource IDs 1000011 or 1000503.

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

3. Add the resources.

Structure of the Web Hosting Plan

The plan contains one product, Web Host Product, with a rate plan, Monthly Cycle Forward Event, and a rate plan selector, Cycle Fold Event.

Monthly Cycle Forward Event rate plan

This rate plan consists of one tier, the Web Host Rate Tier, that charges \$40.00 per month of Web hosting service.

Cycle Fold Event rate plan selector

This rate plan selector includes two rate tiers, a Warning Tier and a Charge For Extra MB tier.

The Warning Tier has two discount brackets. The first charges zero US Dollars if the Peak Megabytes resource balance is between 0 and 101 at the end of the month. The second increases the resource balance of the Warning resource by 1.00 if the Peak Megabytes resource balance is 101 or higher at the end of the month. This sends a warning message to the user that they have exceeded the 100 megabytes of storage offered by the plan.

The Charge for Extra MB Tier also has two discount brackets. The first charges zero US Dollars if the Peak Megabytes resource balance is between 0 and 101 at the end of the month.

The second discount bracket charges 10 cents for each megabyte used if the peak MBs used in a month is over 100. The discount bracket also credits the account \$10, which is the cost of the first 100 megabytes at 10 cents a megabyte. This credit is issued so that you charge only for the megabytes over 100.

The credit limit of the Warning Tier is set at 1.00 in the Web Host Plan. So the Charge for Extra MB Tier does not take effect until that credit limit is reached. This means the first month customers exceed the 100 MB threshold, they receive a warning email but are not charged for the excess usage.

Questions about the Plan

Why is there a quantity bracket tier in each of the Cycle Fold Event tiers that charges \$0?

This is for readability. It is easier to see that there are no extra charges for the first 100 megabytes. You could remove the first tier and simply start at 101 megabytes of usage, but in an environment where price plans are often modified and used by multiple people, it is important to make the structure of the plan as clear as possible.

Why does the second discount bracket issue a credit of \$10? Would it not be easier to code the Used More Disk Space custom event to track the number of megabytes over 100 used in a month?

This design enables you to quickly and easily change the number of megabytes allowed in a month. If you code the Used More Disk Space custom event to track the number of megabytes over 100 used in a month, you must update the code to change

the number of megabytes allowed in a month rather than simply changing the pricing plan.

Part II

Setting Up Pricing

Part II describes how to set up your price list, test it, and migrate old price list data to the new system. It also describes how to configure real-time and pipeline rating.

Part II contains the following chapters:

- [Setting Up Price List Data](#)
- [Working with Provisioning Tags](#)
- [Working with Extended Rating Attributes](#)
- [Rating Implementation and Customization](#)
- [Testing Your Price List](#)
- [Using the XML Pricing Interface to Create a Price List](#)
- [Migrating Price List Data from Legacy Databases](#)
- [About Real-Time Rate Plans](#)
- [Real-Time Rating Based on Multiple RUMs](#)
- [Rating Based on Multiple RUMs with Pipeline Manager](#)
- [Real-Time Rating Based on Date and Time](#)
- [Real-Time Rating Based on Event or Resource Quantity](#)
- [Using Event Attributes to Rate Events in Real Time](#)
- [About Pipeline Rate Plans](#)
- [Setting Up Pipeline Price List Data](#)
- [Rating by Date and Time with Pipeline Manager](#)
- [Setting Up Zones for Batch Pipeline Rating](#)

Setting Up Price List Data

This chapter describes how to set up the Oracle Communications Billing and Revenue Management (BRM) price list data. This data is used by both real-time rating and pipeline batch rating.

Before reading this chapter, read "[About Creating a Price List](#)".

About Configuring Price List Data

Before creating your price list, you must do the following tasks to configure price list data:

- Create resources. See "[Setting Up Resources](#)".
- Set up offer profiles. See "[Setting Up Offer Profiles](#)".
- Create ratable usage metrics (RUMs). See "[Setting Up Ratable Usage Metrics \(RUMs\)](#)".
- Map event types to RUMs. See "[Mapping Event Types to RUMs](#)".
- Specify which events you must rate. See "[Mapping Event Types to Services](#)".

Setting Up Resources

For background information about resources, see "[About Resources](#)".

Real-time rate plans and pipeline rate plans each use a different set of resources. See the following topics:

- [Setting Up Resources for Real-Time Rating](#)
- [Setting Up Pipeline Manager Resources](#)

Important: You must use the same resource names for real-time and batch pipeline resources.

Setting Up Resources for Real-Time Rating

Before you can create a price list, you must use the Resource Editor in Pricing Center to create resources. You can create currency and non-currency resources.

When you create resources, you define the following:

- The resource name, such as US Dollars.

- The resource ID, such as 840. Currency resource IDs are defined in an ISO standard.
- The rounding value. For example, to round US dollars to cents, specify a rounding value of two places after the decimal point.

Note: Resource Editor sets rounding only for A/R actions such as adjustments and refunds. To set rounding for rating, discounting, and taxation as well as A/R, configure rounding in the `pin_beid` file. See ["About Resource Rounding"](#).

- How to round additional numbers beyond the rounding value:
 - **Up:** For example, 10.151 rounds to 10.16.
 - **Down:** For example, 10.159 rounds to 10.15.
 - **Nearest:** If the additional digit is 0-4, the last significant digit remains the same. If the additional digit is 5-9, the last significant digit is rounded up. For example, 10.144 rounds to 10.14 and 10.145 rounds to 10.15.
 - **Even:** If the additional digit is 0-4, the last significant digit remains the same. If the additional digit is 6-9, the last significant digit is rounded up. If the additional digit is 5, the last significant digit is rounded to the nearest even digit. For example, if rounding is set to two significant digits, 10.155 rounds to 10.16 and 10.165 rounds to 10.16.
- For currency resources, you also define the error tolerance and the abbreviations and symbols used for display, such as \$.
- The order in which to consume resource sub-balances. For example, if a customer's balance contains free minutes with different validity periods, you specify which minutes to use first, according to the starting validity date or ending validity date. See ["Specifying the Order in Which Resource Sub-Balances Are Consumed"](#).

Setting Up Pipeline Manager Resources

Pipeline Manager resource IDs must match real-time resource IDs. For example, when defining pipeline rate plans, you use BRM resource IDs, such as US Dollars (840). Pipeline Manager resources can be mapped to real-time resources. Pipeline Manager resources, which can be defined in a variety of ways, can be combined on the real-time side.

Defining Currencies

You must define all valid currencies before setting up currency resources.

You define Pipeline Manager currencies in Pricing Center.

You must also configure the `DAT_Currency` module. This module converts currency symbols to numeric values and retrieves resource rounding rules, using data from `/config/beid` objects in the BRM database.

Defining a Resource

Resources are arbitrary measurement values for which a separate charge can be calculated. You can define two types of resources: **Money** and **Other**. **Money** resources

are based on a defined currency. **Other** resources are non-currency resources, such as duration or loyalty points.

You define Pipeline Manager resources in Pricing Center.

Defining Currency Exchange Rates

You must set up an exchange rate for each currency you support for rating. Exchange rates specify the rate for converting one currency into another.

Call details records (CDRs) can come from multiple networks in different countries, which use different currencies. When you use exchange rates, you can store up to three charge packets with different currencies in a single EDR. The currency types are:

- **Rating currency:** The currency defined in the rate plan and price model.
- **Billing currency:** The currency associated with a specific customer, network operator, and so forth for billing and invoicing.
- **Home currency:** The Pipeline Manager systemwide currency.

You convert currencies from the rating currency to the billing currency, the home currency, or both.

You define exchange rates in Pricing Center. Each exchange rate includes the following data:

- The currency to convert from.
- The currency to convert to.
- The conversion rate.
- The date from which the conversion rate is valid.

To implement currency conversion, you configure the FCT_ExchangeRate module and the DAT_ExchangeRate module.

When you configure FCT_ExchangeRate, you specify the exchange rate conversion mode by setting the **Mode** entry to Normal or Reverse. In Normal mode, the exchange rate is the multiplier to convert the From Currency to the To Currency. In Reverse mode, the exchange rate is the multiplier to convert the From Currency to the To Currency and the divisor to convert the To Currency back to the From Currency.

For example, suppose the exchange rate to convert from EUR to USD is defined as 2.0. One pipeline converts EUR to USD with **Mode** set to Normal and the another pipeline converts USD to EUR with Mode set to Reverse. In Normal mode, the exchange rate 2.0 is the multiplier to convert from EUR to USD ($\text{EUR} * 2.0 = \text{USD}$). In Reverse mode, the exchange rate 2.0 is the divisor to convert from USD to EUR ($\text{USD} / 2.0 = \text{EUR}$). If Reverse mode was not used, you would define another exchange rate 0.50 to convert from USD to EUR.

To update conversion rates, you use the DAT_ExchangeRate module **Reload** semaphore. The new exchange rates overwrite the old rates.

About currency exchange validity dates

You can use the FCT_ExchangeRate module to configure how to define the validity date for currency conversion for two cases.

- Use the **RatingDateBilling** registry entry to convert from the rating currency to the billing currency.
- Use the **RatingDateHome** registry entry to convert from the rating currency to the home currency.

You can specify a different value for the conversion to the billing currency and the home currency. For example, the validity date for converting the rating currency to the billing currency is usually the system time. The validity date for converting the rating date to the home currency is usually the CDR date.

For both registry entries, the values are:

- SYSTEM. The system time.
- FILE. The time that the file that includes the CDR was created (the file header time stamp).
- CDR. The time that the CDR was generated.
- NONE. (default).

Note: If you specify **RatingDateBilling** in addition to the **RatingDateHome** and **HomeCurrency** parameters, it converts the currency to the home currency only.

Mapping Pipeline Manager Currencies and Real-Time Rating Currencies

Currencies are stored as strings for Pipeline Manager and as numbers for real-time rating. You must map Pipeline Manager currency IDs to real-time rating currency names. To do so, you create a list of currencies in the FCT_BillingRecord module registry. See "About Consolidation for BRM Billing" in *BRM Configuring Pipeline Rating and Discounting*.

Setting Up Offer Profiles

Set up each offer profile in the following way:

1. Provide a unique name for the offer profile.

Important: When setting up a product or a discount for an existing offer profile, use the offer profile name as the provision tag of the product or discount.

Alternately, when you create an offer profile for an existing product or discount, use the provisioning tag of the product or discount as the name for the offer profile.

2. Configure a set of policy labels that define the currency or non-currency resource and the quality of service in rate tiers.

For example, you can define a policy label called *Fair Usage* for a non-currency resource (such as *Megabytes Used*, whose resource Id is 100009). Set profile status levels based on predefined quality of service. The gradation may be as follows:

- *Low QoS* for usage between 0 and 2.5 Megabytes
- *Medium QoS* for usage between 2.5 and 5 Megabytes
- *High QoS* for usage above 5 Megabytes

[Example 3-1](#) shows the contents of one sample policy label.

Example 3-1 Sample Policy Label

```

<policy_label>
  <label>Fair Usage</label>
  <resource_name>Megabytes Used</resource_name>
  <resource_id>100012</resource_id>
  <unit> 1 </unit>
  <tier>
    <tier_start_range>0</tier_start_range>
    <tier_end_range>2.5</tier_end_range>
    <status_label>High Qos</status_label>
  </tier>
  <tier>
    <tier_start_range>2.5</tier_start_range>
    <tier_end_range>5</tier_end_range>
    <status_label>Med Qos</status_label>
  </tier>
  <tier>
    <tier_start_range>5</tier_start_range>
    <status_label>Low Qos</status_label>
  </tier>
</policy_label>

```

About Resource Rounding

BRM enables you to create various rounding rules for different resources, events, and processes such as rating and discounting. You create rounding rules for several reasons:

- To increase the accuracy of rating and discounting results.
- To process usage fees more efficiently. For example, an infinite number such as 5.333... is more easily processed when it is rounded.
- To round for various currencies that use a different number of digits to the right of the decimal (for example, 10.25 dollars and 10 yen).
- To comply with currency conversion rules.
- To bill customers an amount that they can actually pay.

You configure rounding by editing the **pin_beid** file and loading the contents of the file into the **/config/beid** object in the BRM database. For more information, see ["Configuring Resource Rounding"](#).

About Rounding Rules

You can configure resource rounding based on the following rounding criteria:

- The *rounding scale* is the number of significant digits to the right of the decimal point. For example, a scale of 2 applied to 10.321111 rounds to 10.32.
- The *rounding mode* defines whether the number is rounded up, down, or not at all, based on the value of the digit following the last significant digit. See ["About Rounding Modes"](#).
- The process to which the rounding configuration applies. You can specify one of these processes: rating, discounting, taxation, and accounts receivable (A/R). A/R includes actions such as billing, payments, adjustments, cancellations, and G/L reporting.

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 3–1 show how a US dollars resource and purchase event combination can be rounded various ways for various processes:

Table 3–1 Currency Resource and Event Type

For a US Dollars Resource	For This Process	Use This Rounding Scale	Use This Rounding Mode
Event type = <code>/event/billing/product/fee/purchase</code>	Rating	6	Down
Event type = <code>/event/billing/product/fee/purchase</code>	Discounting	6	Up
Event type = <code>/event/billing/product/fee/purchase</code>	A/R	2	Nearest
Event type = <code>/event/billing/product/fee/purchase</code>	Taxation	2	Nearest

You can specify any combination of scale and mode for resource, event, and process combinations.

After each process that performs rounding (rating, discounting, and so forth), the balance impact of the event contains the rounded amount.

To configure rounding, you specify the rounding rules in the resource configuration file (`BRM_Home/sys/data/pricing/example/pin_beid`), and then you load the file by using the `load_pin_beid` utility. The `pin_beid` file contains default rounding rules that you can use. For more information, see "[Configuring Resource Rounding](#)".

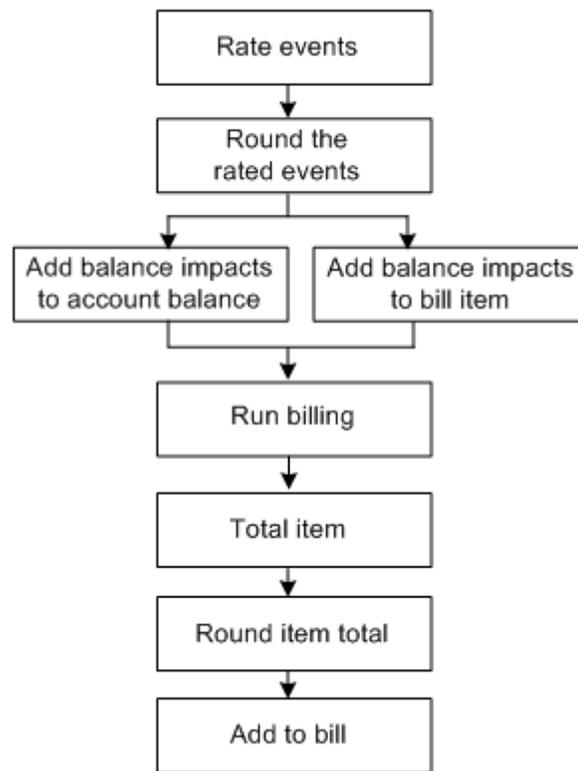
Note: To round non-currency aggregation counter balances for discounting purposes, use Pipeline Manager discount expressions. For more information, see "[About Rounding Aggregation Counter Resources for Discounting](#)".

How BRM Applies Rounding

Rating, discounting, and taxation produce balance impacts, which are rounded and applied to the customer's account balance. The balance impacts are rounded to the scale and mode configured for the event, resource, and process combination.

Balance impacts for an account are stored in bill items, which are associated with the customer's bill. When you run billing, the item totals are rounded according to the A/R rounding rule. Item totals are rounded before being added to the bill so that the bill itself never needs rounding. Before billing, item totals are unrounded and their amounts reflect the scale configured for the associated events.

Figure 3–1 show the general process of balance impact rounding:

Figure 3-1 BRM Balance Impact Rounding

Both real-time rating and pipeline batch rating perform rounding. During pipeline rating, the pipeline rounds the amount in each charge and discount packet.

To round in a pipeline, you configure the Pipeline Manager rounding module. See ["Setting Up Rounding in Pipeline Manager"](#). To configure the rounding rules, see ["Configuring Resource Rounding"](#).

Assuming that rounding is configured for all events and all processes (rating, discounting, taxation, and A/R), rounding is performed in the following order:

1. **Purchase and cycle fees.** When a plan is purchased, the purchase and cycle fees are rounded, if necessary, and a balance impact is generated with the rounded amount. Purchase and cycle fees typically only need rounding when a user purchases or cancels a plan in the middle of the billing cycle. In this case, cycle fees and non-currency grant amounts are prorated, if necessary, and then rounded.
2. **Usage fees.** As customers use their services, the usage events are rated, the usage fees are rounded, and balance impacts are generated with the rounded amounts.
3. **Discount fees.** If a usage discount applies, the discount is calculated on the rounded usage fee, and then the discount is rounded and a discount balance impact is generated with the rounded amount. If there are multiple usage discounts, each discount is rounded in sequence.
4. **Taxes.** If tax should be applied to the event, the tax is calculated on the fee rounded from the previous stage, and then the tax is rounded and a balance impact is generated with the rounded amount.
5. **Billing discounts.** When you run billing, if a billing-time discount applies, it is applied to the total of the rounded usage items, and then the discount is rounded and a discount balance impact is generated with the rounded amount.

6. **Bills.** When the bill is generated, the total amount in each item is rounded, and then all item totals are summed and included in the bill.
7. **Rollover.** If there are any rollover amounts, the rollover is calculated on the total rounded usage, and then the rollover amount is rounded and applied to the next billing cycle. For example, if 100 minutes can be rolled over and the customer used 60.4 minutes, the unused amount of 39.6 minutes is rounded and then added to the account balance for the next cycle.

To round for rollover, you configure a rounding rule for the non-currency resource and cycle forward event combination. Rollover is typically rounded up.

About Rounding and A/R Actions

When entering amounts for A/R actions such as adjustments and refunds, customer service representatives (CSRs) typically use *natural scale*: that is, the scale commonly used in the marketplace for that resource. For example, a person who purchases a book using US dollars cannot make change smaller than one cent (.01 dollars), making 2 the natural scale for US dollars. However, if a CSR reverses or adjusts an event before billing, the scale used is the one in the event’s balance impact. By default, this is the natural scale, unless you change this to use a scale other than natural.

About Rounding Billed and Unbilled Items

The balance impacts of events associated with items can have a high scale. Before billing, item totals reflect the scale of their associated events. During billing, item totals are rounded using the A/R rounding configuration so that customers’ bills display the natural scale for the resource. However, the events associated with billed items still retain their pre-billing scale.

If any operation is performed on billed items: for example, event-level and item-level adjustments, BRM rounds the balance impact of the operation according to the A/R rounding rule to maintain G/L integrity. Because actions on billed items are rounded when they occur, only pending items are rounded when billing is run.

About Rounding for Specific Event Types

When you configure a rounding rule for a specific event type, such as cycle or session events, that rounding rule applies to those events only for the process specified. If you do not configure a rule for every process, all other processes for that event type use the default rounding rule defined in the `pin_beid` file. The default rounding rule specifies natural scale for all events and the rounding mode most commonly used for the process. A rule for all events is specified by using an asterisk (*) as the default event type.

For example, given the configuration in [Table 3–2](#), 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 3–2 Rounding for Specific Event Types

Resource	Event Type	Process	Rounding Scale	Rounding Mode
US dollars	/event/session	Rating	6	Down
US dollars	* (all events)	Taxation	2	Nearest

About Rounding Aggregation Counter Resources for Discounting

For discount rounding, the rounding configurations in the `pin_beid` file are used to round only the balance impacts of discounting, not input balances such as aggregation counters. To round aggregation counter balances in a pipeline, you use pipeline discount expressions when you configure your discounts. To configure aggregation counter rounding, see ["Configuring Rounding Rules for Aggregation Counter Resources"](#).

About G/L Report Rounding

G/L report rounding uses the rounding rule configured for A/R actions. The rounded totals in G/L reports might differ slightly from the total of the rounded bills. This is because item totals are rounded for billing, and journal entries are rounded for G/L reports. Also, G/L reports are rounded differently before and after billing. For more information, see *"About Rounding and G/L Reports"* in *BRM Collecting General Ledger Data*.

If there is any difference between the rounded journal entries in G/L reports and the rounded bill items, BRM records the difference in the G/L. You configure a G/L ID for the rounding difference and configure BRM to record the differences. See ["Configuring to Record Rounding Differences in the G/L"](#).

About Rounding Modes

A rounding mode defines whether a number is rounded up, down, or not at all. The rounding mode rounds to the specified scale. For example, using a scale of 2, rounding up 10.2369 results in 10.24. If the scale is 3, rounding up results in 10.237.

The following rounding mode values that you can use in the `pin_beid` file are defined in the `BRM_Home/include/pin_bill.h` file:

- 0: `PIN_BEID_ROUND_NEAREST`

This mode rounds up or down depending on the value of the digit following the last significant digit. If the additional digit is 0-4, the last significant digit remains the same. If the additional digit is 5-9, the last significant digit is rounded up. For example, if the scale is 2, 10.144 rounds to 10.14 and 10.145 rounds to 10.15. This is the most common rounding method.

- 1: `PIN_BEID_ROUND_UP`

This mode rounds up when the digit following the last significant digit is greater than 0. For example, If the scale is 2, 10.151 rounds to 10.16. If the scale is 1, it rounds to 10.2.

- 2: `PIN_BEID_ROUND_DOWN`

This mode truncates all digits following the last significant digit. For example, if the scale is 2, 10.159 rounds to 10.15. If the scale is 1, it rounds to 10.1.

- 3: `PIN_BEID_ROUND_EVEN`

This mode rounds one of three ways depending on the value of the digit following the last significant digit:

- If it is less than 5, truncate all digits following the last significant digit.
- If it is greater than 5, round up.
- If it is 5, round to the nearest even digit. For example, if the scale is 2, 10.155 and 10.165 both round to 10.16 because 6 is an even number.

- 4: PIN_BEID_ROUND_FLOOR

This mode rounds numbers toward a negative value (that is, the rounded number is always less than the unrounded number). This enables you to round balance impacts so that customers always benefit. For example, if the scale is 2, a credit to a customer of -7.999 is rounded to -8.00, and a debit of 7.999 is rounded to 7.99.
- The following two modes perform the same rounding as their non-alternative counterparts (ROUND_FLOOR and ROUND_DOWN), except that they compensate for possible loss of precision when rounding down by first rounding with a mode of NEAREST using a scale that is two digits greater than the scale you configure.
 - 5: PIN_BEID_ROUND_FLOOR_ALT
 - 6: PIN_BEID_ROUND_DOWN_ALT

For more information, see ["About Rounding Modes That Correct for Loss of Precision"](#).

About Rounding Modes That Correct for Loss of Precision

Some calculations produce results that are slightly less than expected when a value is rounded down. For example, when BRM prorates a \$60.00 cycle fee for 20 out of 30 active days, the calculation is $(20/30) * \$60.00$. The expected result is a fee of \$40.00. However, because $20/30$ evaluates to 0.666..., when this is multiplied by 60 and rounded down, the actual result is a fee of \$39.99.

BRM provides two alternative rounding modes that compensate for possible precision loss when rounding down: ROUND_DOWN_ALT and ROUND_FLOOR_ALT. These modes perform the same rounding as their non-alternative counterparts (ROUND_DOWN and ROUND_FLOOR) after first compensating for loss of precision.

Note: ROUND_DOWN_ALT and ROUND_FLOOR_ALT are not supported in the BRM PCM Java API and in discount expressions.

When these modes are used, if a decimal should be rounded down, BRM performs two rounding functions: The decimal is rounded by using the ROUND_NEAREST rounding mode and a scale that is two more than the scale that you request. It is then rounded down.

For example, if you configure the rounding mode as ROUND_DOWN_ALT and a rounding scale of 2, and the decimal to round is 7.999..., BRM truncates the infinite decimal to the system maximum and then rounds this decimal to the nearest using a scale of 4 (2 more than the configured scale of 2), which results in 8.0000. This decimal is then rounded down using the configured scale of 2, resulting in 8.00 as shown in [Table 3–3](#):

Table 3–3 Rounding Modes That Correct for Precision

Do This	Why
7.999...	Original decimal
7.9999999999999999	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 non-alternative rounding modes. Therefore, the greatest amount that will be modified by using the alternative rounding modes, and still compensate for loss of precision, is less than the greatest amount that will be modified by using the non-alternative rounding modes.

When Rounding Is Not Applied

When the requested rounding scale is greater than the scale of the number being processed, rounding is not required. This occurs when:

- You configure a rounding scale equal to or greater than the maximum number of digits allowed by the system. For example, if the system allows 15 digits and you set the rounding scale to 15 or greater, rounding has no effect.
- You call a rounding function and request a scale that is equal to or greater than the current scale of the decimal: for example, when the decimal is 10.89766 and the scale requested is 5 or greater.
- A computation or expression results in a decimal with a scale that is equal to or less than the configured or requested scale: for example, when a computation results in the decimal 1.98 and the configured scale is 2 or greater.

Configuring Resource Rounding

To set up rounding, perform the following tasks:

- [Configuring Rounding Rules](#)
- [Configuring Rounding Rules for Aggregation Counter Resources](#)
- [Configuring to Record Rounding Differences in the G/L](#)
- [Setting Up Rounding in Pipeline Manager](#)

About Configuring Rounding Rules

You define rounding rules in the balance element ID (BEID) configuration file (*BRM_Home/sys/data/pricing/example/pin_beid*) and then run the `load_pin_beid` utility to load the contents of the file into the `/config/beid` object in the BRM database. See "[Configuring Rounding Rules](#)".

The `pin_beid` file contains default rounding rules based on the most commonly used rounding for the resource and operation. The default rules specify a configuration for all events by using an asterisk (*) as the default event type. For an explanation of the configuration syntax and fields, see the `pin_beid` file.

You configure rounding for resource and event type combinations (for example, US dollars and `/event/session/telco/gsm` events). For each resource and event combination, you specify a rounding scale, a rounding mode, and the process for which this rule applies. For the process, you specify one of these enumerated values in the `pin_beid` file:

- Rating = 0
- Discount = 1
- Taxation = 2

- $A/R = 3$

To configure an event and resource rule for every process, you add an entry for each process.

Important:

- You do not need to configure rounding for all of the processes. However, you must configure rounding for every resource and event type that has a balance impact.
 - If you add resources that are not already in the `pin_beid` file, you should always include a default rounding configuration for that resource that applies to all event types. Any event type that is not explicitly specified for the resource uses this default rounding rule.
 - You can use a single asterisk (*) to denote a default value such as any event type. However, in combination with other characters, you must use valid regular expressions. For example, specifying `/event/*` is incorrect; specifying `/event/(.)*` is correct.
-
-

Tip: You can use regular expressions in the rounding configurations. For example, `/event/session/(.)*` matches all session events.

For more information, see "[Configuring Rounding Rules](#)".

Prioritizing Rounding Rules

BRM applies the first matching rounding rule in the `/config/beid` file. Therefore, if you have more than one rule that matches an event, resource, and process combination, add them in order of priority in the `pin_beid` file. For example, a configuration that matches a specific event type should be added before a configuration that matches all event types.

If there is no rounding rule in the `/config/beid` object that matches the resource, event, and process combination, and no default rule that applies to all events, the event is not rounded.

About Rounding Mode Values

The rounding modes you specify in the `pin_beid` file have corresponding modes in the following API components. However, their values are not all the same:

- **The BRM decimal data type functions.** When BRM invokes a decimal data type function, it converts the rounding mode in the `/config/beid` object into the corresponding rounding parameter used by the decimal data type function. If you call decimal data type functions in custom applications, specify the function's rounding mode, not the BEID rounding mode.
- **The BAS Decimal class.** The `FCT_Rounding` pipeline module converts the rounding mode in the `/config/beid` object into the corresponding rounding mode used by the BAS rounding method in Pipeline Manager. If you use the BAS Decimal rounding method in custom pipeline modules and iScripts, you should include the `FCT_Rounding` module in your pipeline. Otherwise, you will need to convert the rounding mode in the `/config/beid` object into the BAS rounding mode before calling the BAS rounding method.

Table 3–4 shows the BEID rounding modes and the corresponding rounding parameters for the decimal data type functions and the BAS rounding modes:

Table 3–4 BEID Rounding Modes

BEID Rounding Mode (specified in <code>pin_bill.h</code>)	Rounding Mode Parameter for <code>pbo_decimal</code> functions	BAS Decimal Rounding Mode
0: PIN_BEID_ROUND_NEAREST	5: ROUND_HALF_UP	0: PLAIN
1: PIN_BEID_ROUND_UP	1: ROUND_UP	1: UP
2: PIN_BEID_ROUND_DOWN	2: ROUND_DOWN	2: DOWN and TRUNCATE
3: PIN_BEID_ROUND_EVEN	7: ROUND_HALF_EVEN	3: BANKERS
4: PIN_BEID_ROUND_FLOOR	4: ROUND_FLOOR	101: FLOOR
5: PIN_BEID_ROUND_FLOOR_ALT	8: ROUND_FLOOR_ALT	103: FLOOR_ALT
6: PIN_BEID_ROUND_DOWN_ALT	9: ROUND_DOWN_ALT	102: DOWN_ALT
N/A	3: ROUND_CEILING	N/A
N/A	6: ROUND_HALF_DOWN	N/A
N/A	10: ROUND_UNNECESSARY	N/A

Configuring Rounding Rules

To configure rounding rules, you edit the BEID configuration file and then run the `load_pin_beid` utility to load the contents of the file into the `/config/beid` object in the BRM database.

Important: The `load_pin_beid` utility needs a configuration (`pin.conf`) file in the directory from which you run the utility.

To configure resource rounding:

1. Edit the `pin_beid` file in `BRM_Home/sys/data/pricing/example`. The `pin_beid` file includes instructions.

Caution: The `load_pin_beid` utility overwrites the existing resources. If you are updating resources, you cannot load new resources only. You must load complete sets of resources each time you run the `load_pin_beid` utility.

2. Save the `pin_beid` file.
3. Use the following command to run the `load_pin_beid` utility:

```
load_pin_beid pin_beid
```

If you do not run the utility from the directory in which the file is located, you must include the complete path to the file. For example:

```
load_pin_beid BRM_Home/sys/data/pricing/example/pin_beid
```

Tip: If you copy the **pin_beid** file to the directory from which you run the **load_pin_beid** utility, you do not have to specify the path or file name. The file must be named **pin_beid**.

For more information, see "[load_pin_beid](#)".

4. Stop and restart the Connection Manager (CM).

To verify that the network elements were loaded, you can display the **/config/beid** object by using Object Browser, or use the **robj** command with the **testnap** utility.

After loading the **pin_beid** file, functions that calculate fees and discounts use the rounding rules in the **/config/beid** object.

Configuring Rounding Rules for Aggregation Counter Resources

You use Pricing Center to configure rounding rules for aggregation counter resources. When you set up your discount configurations, use discount expressions to specify how to round the resource. Use the following discount expression syntax:

```
round(expression, rounding_scale, rounding_mode)
```

where expression defines the resource balance to round. This can be any discount expression. To round an aggregation balance, use the **Bal** expression. For more information, see the discussion about setting up discounts in the Pricing Center Help.

For example, to round a balance down to two decimal places for an aggregation counter resource with ID 100099, use the following expression:

```
round( Bal(100099), 2, ROUND_DOWN )
```

About Rounding Modes for Discount Expressions

Rounding modes for discount expressions are equivalent to those you specify in the **pin_beid** file, but they have slightly different names. [Table 3–5](#) lists the rounding mode values you can specify in discount expressions and their **pin_beid** file counterpart.

Table 3–5 Rounding Modes for Discount Expressions

Discount Expression Rounding Mode	Rounding Mode Used in the pin_beid File
ROUND_PLAIN	Nearest
ROUND_UP	Up
ROUND_DOWN	Down
ROUND_BANKERS	Even

For a definition of what these modes represent, see "[About Rounding Modes](#)".

Configuring to Record Rounding Differences in the G/L

To record any difference between rounded bill items and the rounded total in the G/L, perform the following tasks:

- [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.

Important: 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
)
```

Caution: 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
```

For more information, see "Loading General Ledger Configuration Data" in *BRM Collecting General Ledger Data*.

Mapping the Rounding G/L ID to an Event

Because the rounding difference is not a rated event, you must map the G/L ID to an event type. G/L ID mapping is defined in the **reasons.locale** file. You can find a sample of this file in the *BRM_Home/sys/msgs/reasoncodes* directory. The sample file is named **reasons.en_US** and contains the following default entry for the rounding G/L ID mapping:

```
DOMAIN = "Others" ;
STR
    EVENT-GLID
        . . .
        /event/journal/epsilon"          1512 ;
    EVENT-GLID-END
```

Note: `/event/journal/epsilon` is a dummy event type used for reference only.

To change the G/L ID for rounding, you must edit and reload the file. The G/L ID you define in the **reasons.locale** and **pin_glid** files must match. See ["Configuring to Record Rounding Differences in the G/L"](#).

To map the G/L ID for rounding to an event, you use the **load_localized_strings** utility to load the contents of the file into the `/config/map_glid` object. When you run the **load_localized_strings** utility, use this command:

```
load_localized_strings reasons.locale
```

Note: If you are loading a localized version of this file, use the correct file extension for your locale.

Caution: 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*.

Configuring BRM to Record Rounding Differences

By default, rounding differences are not recorded in G/L reports. You can enable this feature by modifying a field in the **billing** instance of the `/config/business_params` object.

You modify the `/config/business_params` object using the **pin_bus_params** utility.

To enable BRM to record rounding differences:

1. Use the following command to create an editable XML file from the **billing** instance of the `/config/business_params` object:

```
pin_bus_params -r BusParamsBilling bus_params_billing.xml
```

This command creates the XML file named **bus_params_billing.xml.out** in your working directory. If you do not want this file in your working directory, specify the full path as part of the file name.

2. Search the XML file for the following line:

```
<GenerateJournalEpsilon>disabled</GenerateJournalEpsilon>
```

3. Change **disabled** to **enabled**.

Caution: BRM uses the XML in this file to overwrite the existing **billing** instance of the **/config/business_params** object. If you delete or modify any other parameters in the file, these changes affect the associated aspects of the BRM billing configuration.

4. Save and close the file.
5. Use the following command to load the change into the **/config/business_params** object:

```
pin_bus_params bus_params_billing.xml
```

You should execute this command from the *BRM_Home/sys/data/config* directory, which includes support files used by the utility.

6. Read the object with the **testnap** utility or Object Browser to verify that all fields are correct.
7. Stop and restart the Connection Manager (CM).
8. (Multischema systems only) Run the **pin_multidb** script with the **-R CONFIG** parameter. For more information, see "pin_multidb" in *BRM System Administrator's Guide*.

Setting Up Rounding in Pipeline Manager

Pipeline Manager uses the rounding rules specified in the **/config/beid** object when applying rounding. To round balance impacts in Pipeline Manager, you configure the **FCT_Rounding** module. See "[About Configuring the FCT_Rounding Module](#)".

You can also set up separate rounding rules when defining a price model. See "[About Rounding Rules in Price Models](#)".

Rounding rules can conflict. For more information, see "[Avoiding Rounding Conflicts in Pipeline Manager](#)".

About Configuring the FCT_Rounding Module

The **FCT_Rounding** module finds the rounding rule in the **/config/beid** object based on the event, resource, and process (rating, discounting, or taxation) combination and then rounds the amount in the relevant EDR packet.

Add **FCT_Rounding** to the pipeline after the processing module for which it is rounding:

- **To round for rating**, add this module after the **FCT_RateAdjust** module.
- **To round for taxation**, add this module after the **ISC_TaxCalc** iScript module.
- **To round for real-time discounting**, add this module after the **FCT_Discount** module.
- **To round for batch discounting**, add this module after the **FCT_Discount** module and before the **FCT_ApplyBalance** module.

You specify the process for which this module is rounding in the **Mode** entry of the module registry:

- **Rating** = Round the balance impact of rating.
- **Taxation** = Round the balance impact of taxation.
- **Discounting** = Round the balance impact of discounting.

For more information, see ["Avoiding Rounding Conflicts in Pipeline Manager"](#).

About Rounding Rules in Price Models

You specify one of these rounding methods when defining a price model:

- **None**: Only the rounding rules configured in FCT_Rounding are used.
- **Plain**: Round up if the last significant digit is 5 or greater.
- **Up**: Always round up to the next highest digit.
- **Down**: Always round down to the next lowest digit.
- **Bank**: If the last significant digit is 5, make it even.

If you specify a rounding method other than **None** for a price model, Pipeline Manager will use that rounding method during rating.

For more information, see ["Avoiding Rounding Conflicts in Pipeline Manager"](#).

Avoiding Rounding Conflicts in Pipeline Manager

For Pipeline Manager, it is possible to specify rounding rules in two places that apply to the same event. Pipeline Manager uses different rounding rules at different points in the rating process:

- The rounding method specified for the price model is used by the rating module.
- The rounding method specified for the resource and event is used in the FCT_Rounding module, which is usually run after the FCT_RateAdjust and FCT_Discount modules in the pipeline.

Because rounding can take place at these different points in the pipeline, you can get unexpected results. To avoid conflicts between different rounding rules, you can do one of the following:

- Choose **None** for the rounding method in the price model. If you do this, only the rounding rules configured in FCT_Rounding are used.
- If you select a rounding method other than **None** in the price model, ensure that the selected method is consistent with the rounding method that FCT_Rounding will use.

Rounding Examples

Rounding is performed after rating, discounting, taxation, and A/R actions such as billing and adjustments.

[Table 3–6](#) 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 3–6 Rounding Examples

Action / Process	Calculated Charge	Rounded Charge & Balance Impact	Account Balance
Purchase plan Rate cycle fee	9.95	9.95	9.95
Use service Rate usage	5.23456789	5.23457	15.18456
Discount usage fee 10%	0.523456789	0.52346	14.66111
Tax usage event 3% (after discount)	0.1413333 = 3% * (rounded usage fee - rounded discount)	0.14	14.80111
Run billing Apply billing-time discount of 5% off total usage	0.24250... = 5% * usage item total after A/R rounding (The usage total does not include the cycle fee.)	0.2425	14.55861
Create bill	14.55861 = Total of all items	14.56 (No balance impact)	14.56

Correcting for Precision Loss When Rounding Down

This example shows the results of rounding when you use the ROUND_DOWN_ALT and ROUND_FLOOR_ALT modes. For more information, see ["About Rounding Modes That Correct for Loss of Precision"](#).

The ROUND_DOWN_ALT and ROUND_FLOOR_ALT modes produce different results than ROUND_DOWN and ROUND_FLOOR only when the three digits following the last significant digit are 995 or greater. (The last significant digit is the digit in the decimal place corresponding to the scale: If the scale is 2, the last significant digit in the number 1.23456 is 3.)

For example:

[Table 3–7](#) shows some rounding results of the ROUND_DOWN_ALT and ROUND_FLOOR_ALT modes as compared to their non-alternative rounding counterparts (ROUND_DOWN and ROUND_FLOOR) for various decimal values and rounding scales.

Table 3–7 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

Table 3–7 (Cont.) Rounding Results

Decimal	Scale	Rounding Mode	Rounding Mode	Rounding Mode	Rounding Mode
NA	NA	DOWN	DOWN_ALT	FLOOR	FLOOR_ALT
-12.8999...	1	-12.8	-12.9	-12.8	-12.9
-12.8999...	2	-12.89	-12.900	-12.89	-12.90
-6.9990	2	-6.99	-6.99	-7.00	-7.00
-6.9990	3	-6.999	-6.999	-6.999	-6.999
7.999...	0	7	8	7	8
7.999...	1	7.9	8.0	7.9	8.0
7.999...	2	7.99	8.00	7.99	8.00
-7.999...	0	-7	-8	-8	-8
-7.999...	2	-7.99	-8.00	-8.00	-8.00

Rounding Using Different Modes

The aggregated effects of rounding on the final balance impact is determined by the mode and scale that you configure. The higher the scale, the less effect the rounding mode has on the final balance impact.

For example, [Table 3–8](#) 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 3–8 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.

Tip: If you calculate the discount without rounding the event, the configuration where rating is rounded down and discounting is rounded up returns the most accurate result. Therefore, this is the best mode configuration to use when you discount events.

Modifying a Rounding Rule

To modify a rounding rule, you must change the rule in the `pin_beid` file and reload the file by running the `load_pin_beid` utility. See ["Configuring Resource Rounding"](#).

Setting Up Ratable Usage Metrics (RUMs)

For background information about RUMs, see ["About Ratable Usage Metrics"](#).

Real-time rate plans and pipeline rate plans each use a different set of RUMs. See the following topics:

- [About Setting Up Rums for Real-Time Rating](#)
- [Setting Up Pipeline Ratable Usage Metric \(RUM\) Groups](#)

About Setting Up Rums for Real-Time Rating

Before you create your price list, you must define the RUMs available for rating. When you define RUMs, you define the following RUM attributes:

- The event type that can be rated by using the RUM (for example, Internet usage events). You can specify more than one RUM for an event type.
- The unit of measurement. For example, to rate duration, you might specify seconds or minutes. To measure bytes, you might specify megabytes or kilobytes.
- The RUM name (for example, "Duration," "Pages," or "Bytes"). The RUM name is displayed in Pricing Center.
- How to calculate the quantity. For example, to calculate the length of a session event, you subtract the time the event started from the time that the event ended:

Event start: 7:00 p.m.

Event end: 9:00 p.m.

Event end - Event start = 2 hours

This RUM is defined as follows:

```
/event/session : Duration : PIN_FLD_END_T-PIN_FLD_START_T : second
```

- The RUM name "Duration" appears in Pricing Center.
- The duration is calculated by subtracting the start time from the end time.
- The duration is calculated by using seconds.

To perform calculations, you can use simple arithmetic by using the following operators:

- +
- -
- *
- /
- (
-)

You can use three data types:

- PIN_FLDT_INT

- PIN_FLDT_DECIMAL
- PIN_FLDT_TSTAMP

You can use fields in a substruct, but you cannot use arrays.

Note: The data used for the quantity calculation is stored in event object fields. For example, the event start and end times are stored in PIN_FLD_START_T and PIN_FLD_END_T. You must be familiar with events and event fields to specify quantity calculations.

To define the RUMs, you edit the `pin_rum` file and load it into the BRM database using the `load_pin_rum` utility. See "[Creating Ratable Usage Metrics](#)".

Creating Ratable Usage Metrics

To create RUMs, you edit the `pin_rum` file and then run the `load_pin_rum` utility to load the contents of the file into the `/config/rum` object in the BRM database.

Important: The `load_pin_rum` utility needs a configuration (`pin.conf`) file in the directory from which you run the utility.

To create ratable usage metrics:

1. Edit the `pin_rum` file in `BRM_Home/sys/data/pricing/example`. The `pin_rum` file includes examples and instructions.

Important: A RUM name can include a maximum of 255 characters.

Caution: The `load_pin_rum` utility overwrites existing RUMs. If you are updating RUMs, you cannot load new RUMs only. You must load complete sets of RUMs each time you run the `load_pin_rum` utility.

2. Save and close the `pin_rum` file.
3. Use the following command to run the `load_pin_rum` utility:

```
load_pin_rum pin_rum_file
```

If you do not run the utility from the directory in which the file is located, you must include the complete path to the file. For example:

```
load_pin_rum BRM_Home/sys/data/pricing/example/pin_rum_file
```

For more information, see "[load_pin_rum](#)".

4. Stop and restart the Connection Manager (CM). If necessary, stop and restart Pricing Center.

To verify that the `pin_rum` file was loaded, you can display the `/config/rum` object by using Object Browser, or use the `robj` command with the `testnap` utility.

Important: Fold events cannot use custom RUMs; therefore, do not assign custom RUMs to fold events in any rate plans. Products configured with custom fold RUMs are rated incorrectly.

Setting Up Pipeline Ratable Usage Metric (RUM) Groups

RUMs are ways in which you can measure events, such as how long they last or how big they are. When you create RUMs, you give them names, such as Duration or Volume, and assign them a type, such as Duration, Event, or Normal.

Units of measurement (UoMs) are the names you give to units you will use to measure events, such as seconds or bytes.

A RUM group associates two or more RUMs together and associates a UoM with each RUM in the group.

When you rate events by using a RUM group, a separate charge packet is created for each RUM.

About Defining Units of Measurement (UoMs)

UoMs are the names you give to units you will use to measure events, such as seconds or bytes.

About Defining Ratable Usage Metrics (RUMs)

RUMs are types of event measurements that you define, such as duration or volume.

About Defining a RUM Group

A RUM group groups two or more RUMs together and associates a UoM with each RUM in the group.

Converting Units of Measurement

The UoM of an incoming EDR can be converted into a UoM needed for rating a particular service. For example, an EDR might include the usage amount in seconds, but the service is configured to charge by minutes. The FCT_UoM_Map module converts seconds to minutes, using rounding rules.

To configure UoM mapping:

1. Use Pricing Center to create a UoM map to convert UoMs. When you convert UoMs, you specify the following:
 - The RUM that uses the UoM.
 - The UoM to convert from.
 - The UoM to convert to.
 - The conversion factor. For example, use a factor of 1024 to convert kilobytes into bytes.
 - The rounding rule. The options are:
 - Nearest (N)
 - Always up (U)
 - Always down (D)
2. Configure the FCT_UoM_Map module.

Mapping Event Types to RUMs

Normally, you map event types to RUMs by using Pricing Center. However, you can edit the **pin_usage_map** file and load its content into the BRM database by running the **load_usage_map** utility. By doing so, you update the **/config/usage_map/system** object.

For background information, see "[About Different Types of Rates](#)".

Important: The **load_usage_map** utility requires a configuration (**pin.conf**) file in the directory from which you run the utility.

1. Edit the **pin_usage_map** file in **BRM_Home/sys/data/pricing/example**. The **pin_usage_map** file includes examples and instructions.

Caution: The **load_usage_map** utility overwrites existing usage maps. If you are updating a set of usage maps, you cannot load new usage maps only. You must load complete sets of usage maps each time you run the **load_usage_map** utility.

2. Save the **pin_usage_map** file.
3. Use the following command to run the **load_usage_map** utility:

```
load_usage_map pin_usage_map_file
```

If you are not in the same directory as the **pin_usage_map** file, include the complete path to the file. For example:

```
load_usage_map BRM_Home/sys/data/pricing/example/pin_usage_map_file
```

For more information, see "[load_usage_map](#)".

4. Stop and restart the Connection Manager (CM). If necessary, stop and restart Pricing Center.

To verify that the data loaded correctly, display the **/config/usage_map/system** object by using Object Browser, or use the **robj** command with the **testnap** utility.

Mapping Event Types to Services

To specify which events to rate, you edit the **pin_event_map** file and run the **load_event_map** utility to store the list of ratable events in your database.

For example, to rate a dialup session, the **pin_event_map** file includes this entry:

```
/service/ip: /event/session/dialup: IP Dialup Event start_t
```

This entry specifies that:

- When you create a product for an IP service, you can rate dialup events.
- The name of the event is displayed in Pricing Center as "IP Dialup Event."
- The start time of the event determines which account sub-balance the event applies to. This field is optional and relevant only for session events in which the duration might overlap the sub-balance validity period.

For example, an account has two sub-balances:

- A sub-balance with a resource of free minutes and a validity period that starts at 1:00 p.m. and ends at 2:00 p.m.
- A sub-balance for dollars that is impacted by telephony usage.

The customer makes a call at 1:55 p.m. and ends the call at 2:10 p.m. If you specify the start time in the **pin_event_map** file for the usage event, the customer is allowed to use the free minutes because the call began after the validity start time, and the free minute sub-balance is impacted. However, if you specify the end time, the customer cannot use the free minutes because the call ended after the validity end time, so charges for the minutes used are applied to the dollars sub-balance.

If this field is not present, the start time is used by default. For more information about sub-balances, see "[About Tracking Resources in Account Sub-Balances](#)".

Important: The **load_event_map** utility requires a configuration (**pin.conf**) file in the directory from which you run the utility.

To map event types to services, you edit the **pin_event_map** file and then run the **load_event_map** utility to load the contents of the file into the **/config/event_map** object in the BRM database.

1. Edit the **pin_event_map** file in *BRM_Home/sys/data/pricing/example*. The **pin_event_map** file includes examples and instructions.

Caution: The **load_usage_map** utility overwrites existing usage maps. If you are updating a set of usage maps, you cannot load new usage maps only. You must load complete sets of usage maps each time you run the **load_usage_map** utility.

2. Save the **pin_event_map** file.
3. Use the following command to run the **load_event_map** utility:

```
load_event_map pin_event_map_file
```

If you are not in the same directory as the **pin_event_map** file, include the complete path to the file, for example:

```
load_event_map BRM_Home/sys/data/pricing/example/pin_event_map_file
```

For more information, see "[load_event_map](#)".

4. Stop and restart the Connection Manager (CM). If necessary, stop and restart Pricing Center.

To verify that the **pin_event_map** file was loaded, you can display the **/config/event_map** object by using Object Browser, or use the **robj** command with the **testnap** utility.

About Tracking Resources in Account Sub-Balances

Your customers' account balances are stored in balance groups. A balance group contains a collection of balances for various resources such as currency, minutes, bytes, and frequent flyer miles. Accounts can have multiple balance groups.

Account balances are grouped by the type of resource they track. Each resource can include one or more sub-balances. A resource includes more than one sub-balance when portions of the resource are valid at different times. For example, a resource of

free minutes might include 300 minutes that are valid only for the current month and 1000 free minutes that never expire.

A currency sub-balance can store the balances for multiple services. For example, an account that owns two products that cost \$25.00 per product has a starting currency sub-balance of \$50.00, providing the services are associated with the same balance group and have the same validity period.

Account sub-balances include the following information:

- The start time and end time for which the sub-balance is valid.

Common resources with the same validity periods are stored in the same sub-balance. Resources with unique validity periods are stored in separate sub-balances. For more information, see "[How Resources in Validity-Based Sub-Balances Are Updated](#)".

For information about rounding the start times of cycle and purchase grants to midnight, see "[Configuring Time-Stamp Rounding for Validity Period Start Times](#)".

- The current amount of the sub-balance.
- Consumed reservation (or active reserved amount) which tracks the consumed reservation and is used to set up notifications to the network.

See the discussion on calculating reservation balances in *BRM Telco Integration*.

- The fields in the event record or object (referred to as "contributors") that contribute to how sub-balances are created, updated, and retrieved. For example, to retrieve the total available balance for a specific service, the service object is specified. To deduct free minutes for a phone call, the session object is specified. A separate sub-balance is kept for each unique contributor. See "[About Sub-Balance Contributors](#)".
- Rollover data such as the rollover period and the resource amount that is rolled over, if any. For more information, see "[About Rollovers](#)".
- ID of the product or discount that granted the resource (referred to as "grantor object.")

You can configure sub-balances to track various types of resources and usage. See "[About Configuring Sub-Balances](#)".

About Non-Currency Sub-Balances

A non-currency sub-balance typically has a limited validity period (for example, the period during which free minutes can be used). Non-currency sub-balances can contain various types of resources, such as:

- Free minutes.
- Frequent flyer miles.
- Loyalty points.
- Number of emails or text messages.

A non-currency sub-balance can also keep track of the total resources used for discounts that are shared among several accounts. In this case, the sub-balance acts as a counter to keep track of the total consumed resource.

For information about creating discounts that can be shared, see "Discount Sharing Configuration Example" in *BRM Configuring Pipeline Rating and Discounting*.

When granting a non-currency resource, if a sub-balance already exists, BRM compares the new balance data with the following data in the existing valid sub-balances:

- Contributor
- Grantor object
- Rollover data
- Valid-from date
- Valid-to date

If the data matches, BRM adds the amount to the existing sub-balance; otherwise, it creates a new sub-balance.

For information about configuring sub-balances, see "[About Configuring Sub-Balances](#)".

How Resources in Validity-Based Sub-Balances Are Updated

By default, BRM stores common resources with the same validity periods in the same sub-balance, provided they are associated with the same balance group. (You can create separate balance groups per service in Pricing Center. See "[Tracking Resources by Service](#)".) BRM automatically creates a new sub-balance for resources with a unique validity period, if one does not already exist.

For example, an account owns two services that each include 100 free minutes that are always valid. The account has a balance of 200 free minutes stored in a single sub-balance. When the customer uses free minutes from each service, the free minutes are consumed from the common sub-balance.

Common resources with different validity periods are tracked in separate sub-balances. For example, an account owns two services that each include 100 free minutes. Free minutes for service 1 expire at the end of the month, and free minutes for service 2 expire at the end of the year. Each set of free minutes is stored in a separate sub-balance.

Note: When common, non-currency resources are configured to start on first usage, BRM creates a new sub-balance for each resource whether or not they have the same validity period. See "[About Non-Currency Sub-Balances That Start on First Usage](#)".

You can specify the order in which sub-balances are consumed by setting up resource consumption rules. See "[Specifying the Order in Which Resource Sub-Balances Are Consumed](#)".

You can also limit how validity-based resources such as free minutes are summed by configuring sub-balances. For example, you might want to limit usage of free minutes to a specific service or a specific call session. See "[About Configuring Sub-Balances](#)".

Configuring Time-Stamp Rounding for Validity Period Start Times

You can configure BRM to round time stamps to midnight for resources granted by cycle and purchase events. See the following:

- [Configuring Time-Stamp Rounding for Cycle Grants](#)
- [Configuring Time-Stamp Rounding for Purchase Grants](#)

For more information, see "About Configuring Time-Stamp Rounding" in *BRM Configuring and Running Billing*.

Configuring Time-Stamp Rounding for Cycle Grants

To configure BRM to round time stamps to midnight for the resources granted by cycle events:

1. Open the CM configuration file (*BRM_Home/sys/cm/pin.conf*) in a text editor.
2. Set the **timestamp_rounding** entry to **1**.
3. Save and close the file.

Configuring Time-Stamp Rounding for Purchase Grants

To configure BRM to round time stamps to midnight for the resources granted by purchase events:

1. Open the CM configuration file (*BRM_Home/sys/cm/pin.conf*) in a text editor.
2. Set the **timestamp_rounding** entry to **1**.
3. Save and close the file.
4. Go to the *BRM_Home/sys/data/config* directory, where *BRM_Home* is the directory in which BRM is installed.
5. Run the following command, which creates 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 working directory. To place this file in a different directory, specify the path as part of the file name.

6. Open the **bus_params_rating.xml.out** file.
7. Search for the following line:

```
<TimestampRoundingForPurchaseGrant>disabled</TimestampRoundingForPurchaseGrant>
```
8. Change **disabled** to **enabled**.
9. Save the file as **bus_params_rating.xml**.
10. Run the following command, which loads this change into the appropriate **/config/business_params** object.

```
pin_bus_params PathToWorkingDirectory/bus_params_rating.xml
```

where *PathToWorkingDirectory* is the directory in which **bus_params_rating.xml** resides.

Caution: BRM uses the XML in this file to overwrite the existing **rating** instance of the **/config/business_params** object. If you delete or modify any other parameters in the file, these changes affect the associated aspects of BRM's rating configuration.

Note: To run the command from a different directory, see the description for **pin_bus_params** in *BRM Developer's Guide*.

11. Read the object with the **testnap** utility or the Object Browser to verify that all fields are correct.

See "Using testnap" in *BRM Developer's Guide* for general instructions on using the **testnap** utility. See "Reading Objects by Using Object Browser" in *BRM Developer's Guide* for information on how to use Object Browser.

12. Stop and restart the CM.

13. (Multischema systems only) Run the **pin_multidb** script with the **-R CONFIG** parameter. For more information, see "pin_multidb" in *BRM System Administrator's Guide*.

About Non-Currency Sub-Balances That Start on First Usage

When a non-currency resource is configured to start on first usage (when the customer first consumes the resource balance), BRM always creates a new sub-balance for that resource when it is granted. A new sub-balance is created for each resource even when a product or discount grants multiple first-usage resources of the same type. (For more information about first-usage start times, see "[About Balance Impacts That Become Valid on First Usage](#)".)

For example, a product includes two grants of 100 free minutes. Each grant starts on first usage and ends at the end of the cycle. When the product is purchased, BRM adds two free-minute sub-balances to the account, one for each grant.

Keeping first-usage resources in separate sub-balances enables those balances to have unique validity periods, which are set individually when each resource is consumed. For example, if there are two balances of 100 free minutes each and the customer makes a 30-minute phone call, the validity period of only the balance that was consumed is set when the call is made.

A resource balance that has a first-usage start time will remain available for consumption for as long as the balance is not used. For example, on January 1, a customer purchases a deal that includes 30 free text messages that are valid for 30 days from first usage. The text message balance remains unused in the account until April 18, when the customer sends the first text message. The validity period of the text message balance is then set to start on April 18 and end 30 days later, on May 17.

About Configuring Sub-Balances

You can optionally configure sub-balances to track resources for more specific types of usage such as:

- Free minutes per call session.
- Frequent flyer miles per service instance.
- Friends and family calls to specific locations.
- Discount amounts shared with sponsored accounts.

You configure sub-balances for the entire BRM system by editing and loading a configuration file.

By default, common resources for all services owned by an account are placed in the same sub-balance if they have the same validity period and are associated with the same balance group. (You can create separate balance groups per service in Pricing Center. See "[Tracking Resources by Service](#)".)

To limit the allocation and consumption of resources, you configure sub-balances by specifying the following values:

- *Resource ID*. This is the ID for the type of resource in the sub-balance, such as dollars or free minutes.
- *Event type*. This is the type of event that impacts the sub-balance, such as GSM usage events (*/event/session/telco/gsm*).
- *Contributors*. Contributors can be any field in the event record or object. There are two types of contributors:
 - *Retrieving contributor*. All sub-balances with common retrieving contributors are summed when sub-balances are retrieved. For example, to retrieve the total balance for specific services, specify the service object as the retrieving contributor.
 - *Updating contributor*. A sub-balance is created or updated by balance impacts for each unique updating contributor. For example, to add or deduct free minutes for specific dialup sessions, specify the session object as the updating contributor.

About Sub-Balance Contributors

Sub-balance contributors are specified by a field name from the event record or object. The value in the field you specify is used to retrieve and update the sub-balances. Some fields you might want to use as contributors include:

- The service object.

Specify a service field to track resources for specific service instances such as fax, telephony, and text messaging.

Note: Specifying the service object in the configuration is one way of creating service-level balances. The other way is to create a balance group for the service when you define your plans in Pricing Center. See "[Tracking Resources by Service](#)".

- The session object.

Specify a session field to track resources per session instance. This is useful when you offer discounts for certain levels of usage (for example, 10 frequent flyer miles per hour of phone calls).
- The product object.

When several products in the same plan have different rollover rules, tracking resources per product permits BRM to update the free minutes for each product.
- The phone number field.

Specify a phone number field to track resources for called telephone numbers.
- The account object.

Specify the account field to track resources that are shared among several accounts, such as earned free minutes. When free minutes are distributed, they can be divided based on each account's usage level.

Retrieving and Updating Sub-Balances

Sub-balances are retrieved for a given resource. For example, when CSRs use Customer Center to view an account's current cash balance, all valid cash sub-balances

(sub-balances with current validity periods and matching contributors) are summed to provide the current resource balance.

Sub-balances are updated by a balance impact. When a balance impact affects more than one sub-balance, the sub-balances are updated in the following order:

1. **By validity period.** By default, sub-balances are updated in chronological order based on the validity start date or end date. You specify the order in which sub-balances are used by setting up resource consumption rules. See "[Specifying the Order in Which Resource Sub-Balances Are Consumed](#)".
2. **By contributor.** Sub-balances with specific field contributors are impacted before sub-balances that accept all contributors (with an asterisk or empty contributor values). See "[Configuring Sub-Balances](#)".

Configuring Sub-Balances

To configure sub-balances, you edit the sub-balance configuration file and then run the `load_pin_sub_bal_contributor` utility to load the contents of the file into the `/config/sub_bal_contributor` object in the BRM database.

Editing the `pin_sub_bal_contributor` File

Each line in the `pin_sub_bal_contributor` file defines the usage for which a sub-balance is created. The configurations apply to all sub-balances in the BRM database, but they are implemented at the balance group level. That is, when a usage event occurs, if the account has more than one balance group, only the sub-balances within the specified balance groups are impacted.

To add sub-balance configurations, use this syntax:

```
resource_type:event_type:retrieving_contributor:updating_contributor
```

For example:

```
1000003:/event/session/gprs:PIN_FLD_SESSION_OBJ:PIN_FLD_SESSION_OBJ
```

where:

- **1000003** is the resource type for frequent flyer miles.
- **/event/session/gprs** is the event type that impacts the frequent flyer miles resource.
- **PIN_FLD_SESSION_OBJ** is the retrieving contributor. Therefore, a separate balance summary is retrieved for each unique GPRS session.
- **PIN_FLD_SESSION_OBJ** is the updating contributor. Therefore, separate sub-balances are created or updated for each unique GPRS session.

The contributors can be a specific field or an asterisk (*) to indicate any contributor. For example, the following entry retrieves and tracks dollars (resource type 840) in a single sub-balance for all events with any contributor:

```
840:/event:*:*
```

To configure several sub-balances for one resource type, leave subsequent resource fields empty. If no resource is specified, the previous resource specified is used. For example, the next two configurations use the same resource (100002):

```
100002 : /event/session/telco/gsm : PIN_FLD_SESSION_OBJ : PIN_FLD_SESSION_OBJ
      : /event/session/dialup : PIN_FLD_SERVICE_OBJ : PIN_FLD_SERVICE_OBJ
```

Running the `load_pin_sub_bal_contributor` Utility

Caution: 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.

Important: The resources specified in the `pin_sub_bal_contributor` file must exist in the BRM database before the `load_pin_sub_bal_contributor` utility is run. Therefore, you must first load the `pin_beid` file by running the `load_pin_beid` utility. See "[load_pin_beid](#)".

Use the following command to run the `load_pin_sub_bal_contributor` utility:

```
load_pin_sub_bal_contributor pin_sub_bal_contributor
```

If you are not in the same directory as the `pin_sub_bal_contributor` file, include the complete path to the file. For example:

```
load_pin_sub_bal_contributor BRM_Home/sys/data/pricing/example/pin_sub_bal_contributor
```

For more information, see "[load_pin_sub_bal_contributor](#)".

To verify that the sub-balance configurations loaded correctly, display the `/config/sub_bal_contributor` object by using Object Browser, or use the `robj` command with the `testnap` utility.

Sub-Balance Configuration Example

The following examples show how sub-balances can be configured for various business needs.

Sub-Balances per Service Example

There are two ways to track resources per service instance:

- By creating a balance group for the service. See "[Tracking Resources by Service](#)".
- By specifying the service object as a contributor in the `pin_sub_bal_contributor` file.

[Table 3–9](#) shows contributor configurations to track resources for GSM services. The resources are dollars and free minutes.

Table 3–9 Contributor Configurations

Resource	Event Type	Retrieving Contributor	Updating Contributor
Dollars	/event/session/gsm	PIN_FLD_SERVICE_OBJ	PIN_FLD_SERVICE_OBJ
Free minutes	/event/session/gsm	PIN_FLD_SERVICE_OBJ	PIN_FLD_SERVICE_OBJ
Free minutes	/event/cycle_forward	PIN_FLD_SERVICE_OBJ	PIN_FLD_SERVICE_OBJ

Table 3–9 (Cont.) Contributor Configurations

Resource	Event Type	Retrieving Contributor	Updating Contributor
Free minutes	/event/cycle_forward	PIN_FLD_SERVICE_OBJ	PIN_FLD_SERVICE_OBJ

Each line in the above example specifies a configuration.

- **Dollars** is a currency sub-balance that is created in the account balance group.
- **/event/session/gsm** is the event type that will impact the dollars resource.
- **PIN_FLD_SERVICE_OBJ** is the retrieving contributor, so the dollars resource for each unique GSM service is summed when retrieving balance information.

If you specify an asterisk instead of **PIN_FLD_SERVICE_OBJ** for the retrieving contributor, dollars for *all* GSM services owned by the account would be summed when retrieving balance information.

- **PIN_FLD_SERVICE_OBJ** is the updating contributor, so a separate sub-balance is created to track dollars for every unique GSM service that the account owns, such as telephony, fax, and SMS.
- **Free minutes** is a non-currency sub-balance that is created in the account balance group.
- **/event/session/gsm** is the event type that will impact the free minutes resource.
- **PIN_FLD_SERVICE_OBJ** is the retrieving contributor, so the free minutes resource for each unique GSM service is summed when retrieving balance information.

If you specify an asterisk instead of **PIN_FLD_SERVICE_OBJ** for the retrieving contributor, free minutes for *all* GSM services owned by the account would be summed when retrieving balance information.

- **PIN_FLD_SERVICE_OBJ** is the updating contributor, so a separate sub-balance is created to track free minutes for every unique GSM service that includes free minutes. Additional free minutes and consumption of free minutes are restricted to the GSM service instance.

If you specify an asterisk instead of **PIN_FLD_SERVICE_OBJ** as the updating contributor, free minutes granted by every GSM service owned by the account would be shared by all the GSM services.

- **Free minutes** is a non-currency sub-balance that is created in the account balance group.
- **/event/cycle_forward** is the event type that will impact the free minutes resource.
- **PIN_FLD_SERVICE_OBJ** is the retrieving contributor, so free minutes granted by cycle forward events is summed for each unique service when retrieving balance information.
- **PIN_FLD_SERVICE_OBJ** is the updating contributor, so a separate sub-balance is created to track free minutes granted by cycle forward events for each unique service that grants free minutes. Consumption of free minutes granted at the beginning of the cycle is restricted to the service that granted them.

Specifying the Order in Which Resource Sub-Balances Are Consumed

Customers are sometimes granted multiple sub-balances of a particular resource, such as minutes. For example, a customer's balance might include minutes granted at the start of the accounting cycle and rollover minutes from the previous month. Because the minutes have different validity periods, they are grouped into different resource sub-balances. See "[About Tracking Resources in Account Sub-Balances](#)".

When the customer uses a service, BRM needs to know which minutes (or sub-balance) to use first. You use resource consumption rules to specify the order in which resource sub-balances are consumed, according to the validity start time and end time.

For example, to use rollover minutes first, you configure BRM to consume sub-balances based on the earliest validity start time. To use the minutes that expire first, you configure BRM to consume sub-balances based on the earliest validity end time.

Consumption rule descriptions

BRM supports the resource consumption rules shown in [Table 3-10](#):

Table 3-10 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 (LETTEST)	Consume the sub-balance with the latest validity end time first. If multiple sub-balances have the same validity end time, use the one with the earliest validity start time first.
Latest expiration time and latest start time (LETLST)	Consume the sub-balance with the latest validity end time first. If multiple sub-balances have the same validity end time, use the one with the latest validity start time first.

For example, if a customer's balance includes the following minutes and the customer makes a phone call on February 10, the consumption rules specify which group of minutes are impacted first:

- 100 Anytime Minutes with a validity period of February 1 to February 28.
- 50 rollover Anytime Minutes with a validity period of January 1 to February 28.
- 200 bonus Anytime Minutes with a validity period of January 15 to June 15.

If the consumption rule is set to earliest start time (EST), BRM applies the balance impact to the 50 rollover Anytime Minutes first. Likewise, if the rule is set to earliest expiration time and latest start time (EETLST), BRM applies the balance impact to the 100 Anytime Minutes first.

You specify the order in which resource sub-balances are consumed in your price plans. You can also set systemwide and default resource settings. BRM reads and uses the consumption rule settings in the order shown below:

1. **Price plan setting.** Your price plans can include resource-to-consumption rule mappings for each service that you support. This enables you to have different consumption rules for the same resource based on the plans owned by the customer. When a customer purchases a plan, the plan's resource-to-consumption rule mapping is stored in the customer's **/balance_group** object. If there are any conflicting rules for the same resource, BRM uses the rule from the most recently purchased plan.
2. **Systemwide resource setting.** You can specify a systemwide resource-to-consumption rule mapping for each service that you support. BRM uses the systemwide settings only if a rule is not defined for the resource in the customer's purchased plans. BRM stores systemwide settings in the **/config/beid** object.
3. **Default resource setting.** You can specify a default setting that applies to all resources. This setting is used only if a consumption rule is not defined for the resource in the customer's purchased plans or if there is not a systemwide resource setting. BRM stores the default resource setting in the **/config/business_params** object.

How BRM Applies Consumption Rules

To apply resource consumption rules, BRM automatically orders an account's resource sub-balances whenever the account adds or changes a resource. For example, if you modify a resource with a consumption rule of EET, BRM orders the resource sub-balances based on the validity end time, starting with the earliest expiration time first. When the account has a balance impact, BRM searches through the account sub-balances, in order, and applies the balance impact to the first sub-balance that matches the following criteria:

- Has a validity period that matches the event's time stamp.
- Has a balance to consume.

If there is any remaining balance impact, BRM applies it to the next sub-balance that matches the criteria. This process continues until all of the sub-balances have been depleted. BRM then restarts the search from the beginning and applies any remaining balance impact to the first sub-balance that matches the validity period.

For example, assume that an account with the following resource sub-balances makes a 30-minute phone call on June 4. If the consumption rule is set to LSTEET, BRM first consumes the 5 minutes from sub-balance A and then consumes the 10 minutes from

sub-balance C. Because the account's sub-balances have been depleted, BRM charges the remaining 15 minutes to sub-balance A.

- Sub-balance A has a balance of 5 minutes with a validity period of June 1 to June 15.
- Sub-balance B has a balance of 0 minutes with a validity period of June 1 to June 30.
- Sub-balance C has a balance of 10 minutes with a validity period of May 1 to July 15.
- Sub-balance D has a balance of 0 minutes with a validity period of January 1 to December 30.

The method BRM uses to implement resource consumption rules is different for batch rating and real-time rating.

How Batch Rating Applies Consumption Rules

In batch rating, resource consumption rules are applied by the DAT_BalanceBatch module. The DAT_BalanceBatch module orders resource sub-balances when Pipeline Manager starts and when an account adds or modifies a resource. When processing CDRs, FCT_ApplyBalance uses the DAT_BalanceBatch module to search through an account's resource sub-balances, in order, and find the first sub-balance that matches the CDR time stamp and has a balance to consume.

By default, Pipeline Manager supports all of your price plan, systemwide, and default consumption rule settings. However, you can configure Pipeline Manager to use the default setting only by using the **UseFlexibleConsumptionRule** registry entry in the DAT_BalanceBatch module:

- **True:** Uses the consumption rules defined for each resource in a balance group. If a consumption rule is not defined, it uses the rules defined in the **/config/beid** object. If a consumption rule is not defined in a balance group or in the **/config/beid** object, the module uses the rule defined in the **multi_bal** instance of the **/config/business_params** object.
- **False:** Uses the systemwide consumption rule defined in the **multi_bal** instance of the **/config/business_params** object only.

How Real-Time Rating Applies Consumption Rules

In real-time rating, consumption rules are applied by the Balance FM opcodes. When an account is created or modified, the Balance FM opcodes read the consumption rule for each resource and order the resource sub-balances appropriately. When a customer has a balance impact, the Balance FM opcodes search through the customer's resource sub-balances, in order, and apply the balance impact to the first sub-balance that matches the event time stamp and has a balance to consume.

Setting Resource Consumption Rules

To assign resource consumption rules, perform one or more of these tasks:

- [Setting Consumption Rules in Your Price Plans](#)
- [Setting Systemwide Consumption Rules for Each Resource](#)
- [Setting the Default Consumption Rule](#)

For information about changing your consumption rule settings, see "[Modifying Resource Consumption Rules](#)".

Setting Consumption Rules in Your Price Plans

You set resource consumption rules in your price plans by using Pricing Center. See the Pricing Center Help for more information.

Note: You can also use the "loadpricelist" utility to assign resource consumption rules in your price plans. You specify the rules in the `price_list.xml` file's `consumption_rule` field and then load the file into the database by using the `loadpricelist` utility. See "Using the XML Pricing Interface to Create a Price List".

Pricing Center and `loadpricelist` store your price lists and consumption rule settings in `/plan` objects in the BRM database. When a customer purchases a product, the plan's resource-to-consumption rule mapping is stored in the customer's `/balance_group` object.

If you use a custom client application to create price lists, you must modify your application to pass consumption rule settings to the Pricing FM opcodes. For more information, see "Customizing Credit Limits and Resource Consumption Rules" in *BRM Managing Customers*.

Setting Systemwide Consumption Rules for Each Resource

You define systemwide consumption rules for each resource that you support by using Resource Editor. See the Resource Editor Help for more information.

Note: You can also use the `pin_beid` configuration file to assign systemwide resource consumption rules. You specify the rules in the file's `con_rule_id` column and then load the file into the database by using the `load_pin_beid` utility. See "load_pin_beid" for more information.

Resource Editor and `load_pin_beid` store the systemwide consumption rule setting for each resource in the `/config/beid` object.

Setting the Default Consumption Rule

The default consumption rule applies to all resources in your system. The default setting is earliest start time and earliest expiration time (ESTEET). You can change this setting by modifying a field in the `multi_bal` instance of the `/config/business_params` object created during BRM installation.

You modify the `/config/business_params` object by using the `pin_bus_params` utility.

To set the default resource consumption rule:

1. Use the following command to create an editable XML file for the `multi_bal` class:

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

This command creates the XML file named `bus_params_multi_bal.xml.out` in your working directory. If you do not want this file in your working directory, specify the path as part of the file name.

2. Search the XML file for following line:

```
<SortValidityBy>ESTEET</SortValidityBy>
```

3. Change **ESTEET** to the appropriate value:
 - Earliest start time (EST)
 - Latest start time (LST)
 - Earliest expiration time (EET)
 - Latest expiration time (LET)
 - Earliest start time and latest expiration time (ESTLET)
 - Earliest start time and earliest expiration time (ESTEET)
 - Latest start time and earliest expiration time (LSTEET)
 - Latest start time and latest expiration time (LSTLET)
 - Earliest expiration time and earliest start time (EETEST)
 - Earliest expiration time and latest start time (EETLST)
 - Latest expiration time and earliest start time (LETEST)
 - Latest expiration time and latest start time (LETLST)

For a description of each setting, see "[Consumption rule descriptions](#)".

Caution: BRM uses the XML in this file to overwrite the existing **multi_bal** instance of the **/config/business_params** object. If you delete or modify any other parameters in the file, these changes affect the associated aspects of the BRM billing configuration.

4. Use the following command to load the change into the **/config/business_params** object:

```
pin_bus_params bus_params_multi_bal.xml
```

You should execute this command from the *BRM_Home/sys/data/config* directory, which includes support files used by the utility.

5. Read the object with the **testnap** utility or Object Browser to verify that all fields are correct.
6. Stop and restart the Connection Manager (CM).
7. (Multischema systems only) Run the **pin_multidb** script with the **-R CONFIG** parameter. For more information, see "pin_multidb" in *BRM System Administrator's Guide*.

Modifying Resource Consumption Rules

You can modify the resource consumption rules that you set in your price plans, in your **/config/beid** object, and in your **/config/business_params** object at any time. However, the new and changed settings apply to new customers only. Your existing customers will continue to use the previous settings:

- Changes to plan-level consumption rules do not trigger any update to existing customers that own the changed plan. Existing plan owners continue to use the plan's previous settings.
- Changes to the **/config/beid** object do not trigger changes to existing customers. Existing customers continue to use the previous systemwide settings.

- Changes to the `/config/business_params` object do not trigger changes to existing customers. Existing customers continue to use the previous default setting.

Important: Pipeline Manager cannot use the new or modified `/config/beid` or `/config/business_params` consumption rule settings until they are loaded into pipeline memory. See "Reloading Data into a Pipeline Manager Module" in *BRM System Administrator's Guide*.

About Rollovers

Use the rollover feature to specify the quantity of unused resources that can be rolled over into subsequent cycles.

For example, you can configure a product so that a portion of unused free minutes from each cycle can be rolled over for use in the following cycle or cycles.

Note: Rollover resources are distinct from resources issued through one-time grants, such as those issued by a CSR or a discount product. One-time grants are valid between two specific dates.

You set up rollovers in a product's event map in Pricing Center.

By default, rollover events are included in the `pin_event_map` file.

About Rollover Resource Sub-Balances

BRM maintains each rollover resource as a sub-balance. Each sub-balance specifies a resource balance that is valid (available for use) between valid-from and valid-to dates stored in BRM. The total amount of a resource available to a customer for each cycle is equal to the sum of all resource sub-balances that are valid during the cycle.

BRM validates whether a sub-balance can be rolled over by checking the rules that govern rollover (such as the amount to roll over, the maximum rollover amount allowed, the maximum number of cycles to roll over, and so on).

When a rollover event occurs, one of three things happens to the balance in a currently valid resource sub-balance:

- **If the full amount is eligible for rollover**, BRM does one of the following:
 - Creates a new rollover sub-balance for the rolled over amount. The rollover sub-balance validity period has the same valid-from date as the original sub-balance, and its valid-to date is extended to the end of the new cycle.
 - Adds the rolled over amount to an existing sub-balance if the existing sub-balance has the same data as would the new rollover sub-balance (such as the resource type, rollover rules, valid-from and valid-to dates, sub-balance contributors, and so on).
- **If only a portion of the resources is eligible for rollover**, the amount in the sub-balance is divided into a non-rollover sub-balance and a rollover sub-balance. The non-rollover sub-balance has the same valid-to date as the original sub-balance. Its resource balance is used for late-arriving usage events. The valid-to date for the rollover sub-balance is extended to the end of the new cycle. Its resources are available for the customer to use in the new cycle.

- **If none of the resource is eligible for rollover**, the sub-balance is not changed. This condition occurs when the sub-balance has already been rolled over the maximum number of times allowed. Amounts in this sub-balance are available only for late-arriving usage events.

Important: Your system's memory limits might limit the number of months you can roll over a sub-balance. The number of resource sub-balances that BRM must maintain varies according to the number of rollover cycles allowed. You set the maximum allowed rollover cycles when configuring the rollover balance impact in Pricing Center.

CSRs can view rollover resource balances by using Customer Center.

When Rollover Events Occur

Resource sub-balances can be rolled over to another cycle as soon as they surpass the valid-to date (expire). Most resource sub-balances expire on the account's billing day of month (DOM). However, some sub-balances may expire in the middle of a cycle due to a product or service cancellation or due to flexible cycles.

Because many BRM features depend on sub-balances being rolled over the day after they expire, BRM rolls over resources at the following times:

- **At the end of the billing cycle:** BRM automatically rolls over all eligible resource sub-balances to the next cycle as part of the billing process. This catches all resource sub-balances that expire on the account's billing DOM.
- **When you run the `pin_bill_day` script:** The `pin_bill_day` script automatically executes the `pin_rollover` utility as part of the billing process. The `pin_rollover` utility rolls over any resource sub-balances that have expired but have not been rolled over to the next cycle. This catches any resource sub-balances that expired in the middle of the cycle.

Deleting Expired Sub-Balances

To delete expired sub-balances, use the `pin_sub_balance_cleanup` utility.

Rollover Example

The following example demonstrates:

- Granting resources (free minutes) at the start of each cycle (calendar month).
- Rolling over unused free minutes from previous months at the start of each month.

Note: For information about rolling over resources that expire in the middle of a cycle, see "[About Rolling Over Resources That Expire in Midcycle](#)".

- Dividing an initial resource grant into rollover and non-rollover sub-balances.
- Limiting the total number of free minutes that can be rolled over from previous months.

- Dividing a rollover balance into rollover and non-rollover sub-balances when the Maximum Cumulative Rollover Total value is reached.
- The default order in which free minutes are consumed from the valid resource sub-balances.
- Expiration of free minutes in a sub-balance when the sub-balance has been rolled over the number of cycles specified by the Maximum Number of Rollover Cycles value.

In this example, the customer’s product includes:

- 500 free minutes granted at the beginning of each usage cycle.
- A rollover specifying that:
 - Up to 100 unused free minutes from each month can be rolled over.
 - The maximum number of rollover cycles is 2.
 - A limit of 150 *total* rollover minutes from previous months can be rolled over into a new month.

Other assumptions:

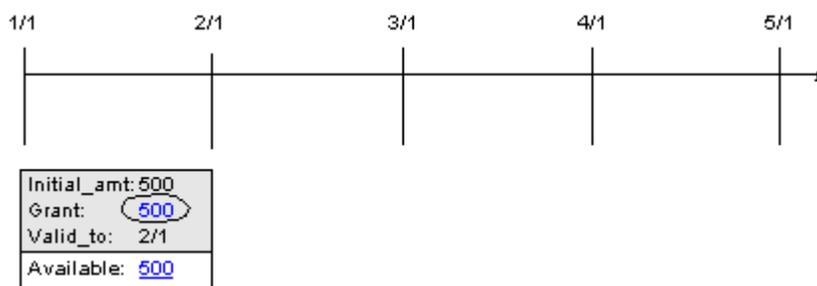
- Cycles are monthly and start on the first day of each calendar month.
- The product with the rollover is valid starting January 1.
- The customer consumes no free minutes until March.

Note: In this example:

- New sub-balances, or those that have a change in their validity period, are highlighted in gray.
 - New grant resource values or changes to current resource amounts within the sub-balances are in **bold black**.
 - Components of available resource totals are in **blue**.
 - Components of used resource totals are in **red**.
-

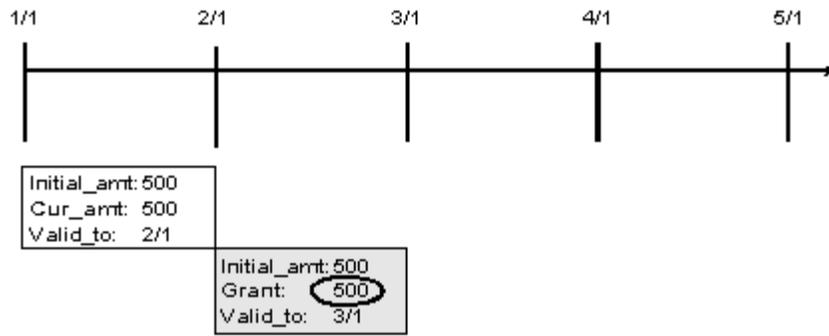
1. On January 1, the customer is granted 500 free minutes for use during January as shown in [Figure 3-2](#). The free minutes are maintained in a resource sub-balance.

Figure 3-2 Initial 500 Free Minutes Grant in January



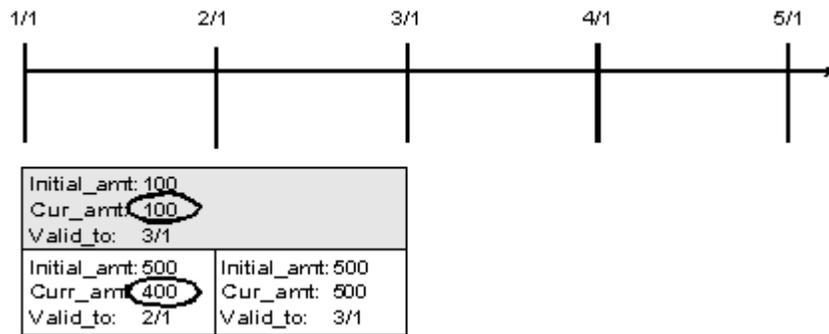
2. On February 1:
 - The cycle forward event creates a new resource sub-balance to track the grant of 500 free minutes for use during February as shown in [Figure 3-3](#).

Figure 3–3 February Free Minutes Grant



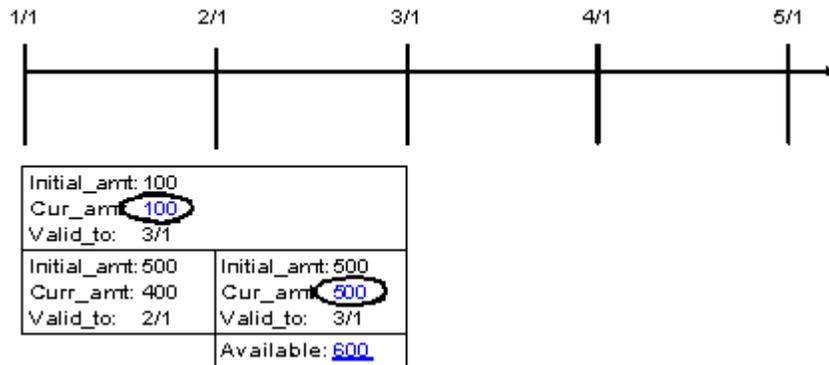
- The cycle rollover event creates a new resource sub-balance for tracking unused January free minutes that roll over into February. 100 minutes are put in the new sub-balance, and they are valid until March 1. The original sub-balance for January is decremented 100 minutes, leaving 400 minutes available for late-arriving January events as shown in [Figure 3–4](#).

Figure 3–4 January Free Minutes Rollover



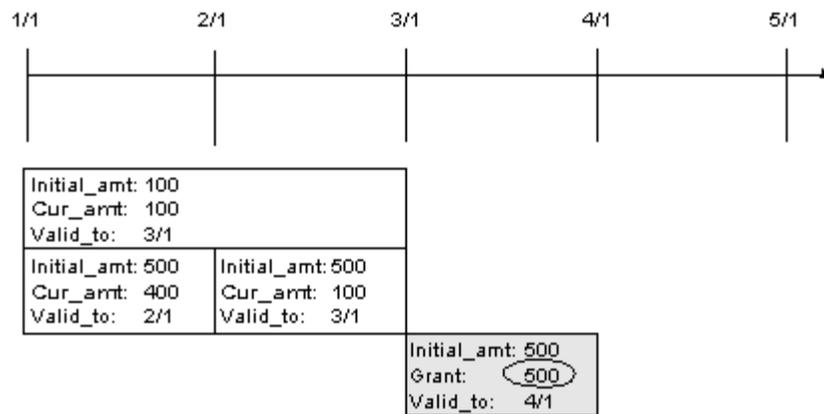
The customer now has 600 free minutes (500 granted February 1 plus 100 rolled over from January) available for use during February as shown in [Figure 3–5](#).

Figure 3–5 February Total Free Minutes Available



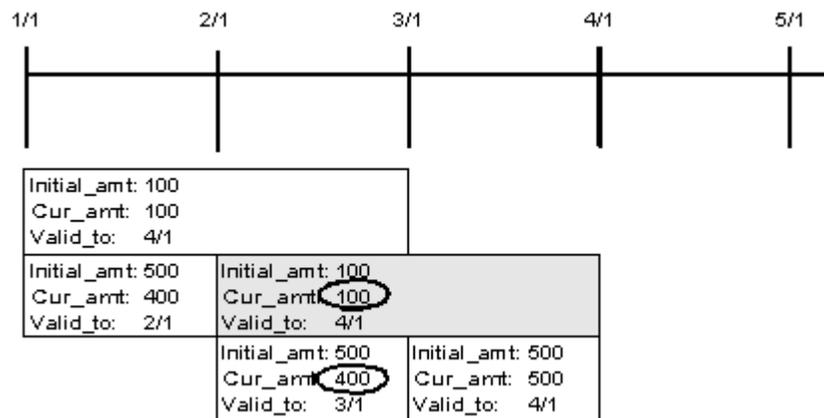
- On March 1:
 - The cycle forward event creates a new resource sub-balance to track the grant of 500 free minutes for use during March as shown in [Figure 3–6](#).

Figure 3–6 March Free Minutes Grant



- The cycle rollover event creates a new resource sub-balance for tracking unused February free minutes that roll over into March as shown in [Figure 3–7](#). 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.

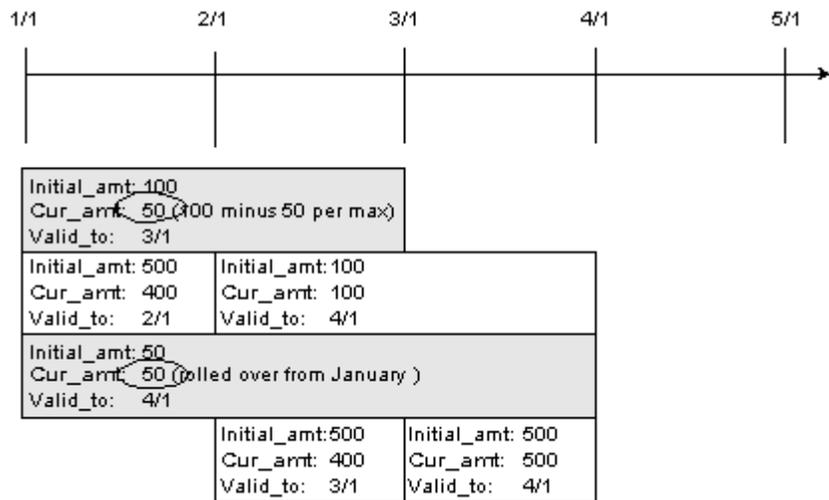
Figure 3–7 February Free Minutes Rollover



- The cycle rollover event divides the January rollover minutes into two sub-balances to enforce the rule that only 150 *total* rollover minutes for a resource can carry forward into a new month. Because 100 free February minutes are already rolled over, only 50 minutes can be rolled over from the January rollover sub-balance for use in March.

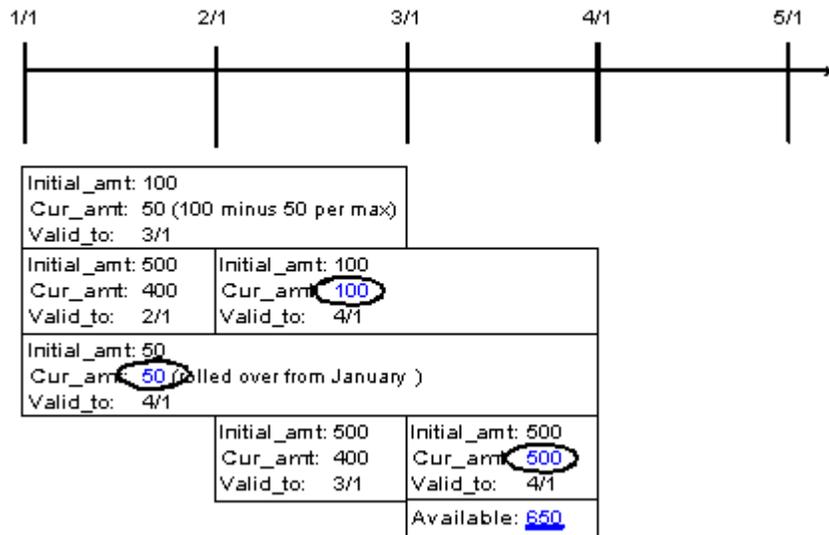
One new sub-balance contains 50 minutes available for use for late-arriving February and January events. The other contains 50 rollover minutes that the customer can use in March. The resource balances for each month are shown in [Figure 3–8](#).

Figure 3–8 Remaining January Free Minutes Rollover



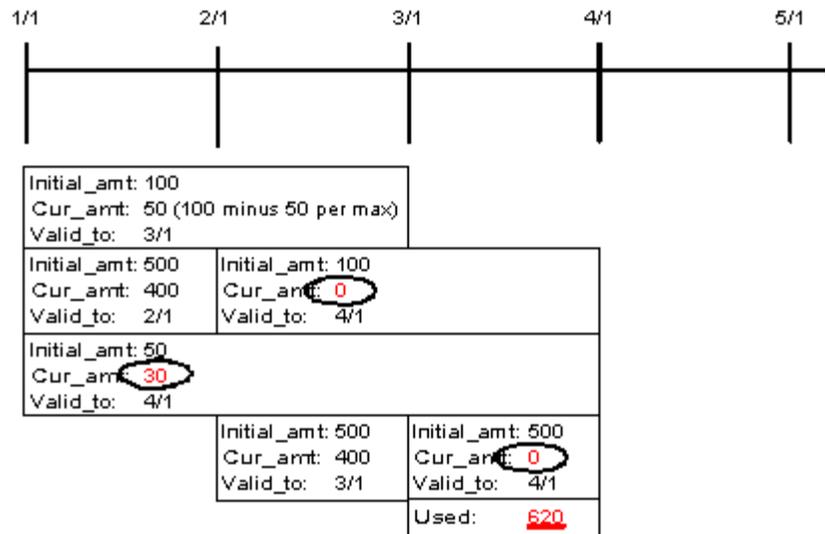
The customer now has 650 free minutes (500 granted March 1 plus 100 rolled over from February plus 50 rolled over from January) available for use during March as shown in Figure 3–9.

Figure 3–9 March Total Free Minutes Available



4. During March, the customer consumes 620 minutes as shown in Figure 3–10. Sub-balance resources are used starting with the newest sub-balance, as indicated by the sub-balance valid-from dates:
 - All 500 free minutes from the March grant are consumed, leaving a zero balance.
 - All 100 February rollover minutes are consumed, leaving a zero balance.
 - 20 of the 50 January rollover minutes are consumed, leaving 30.

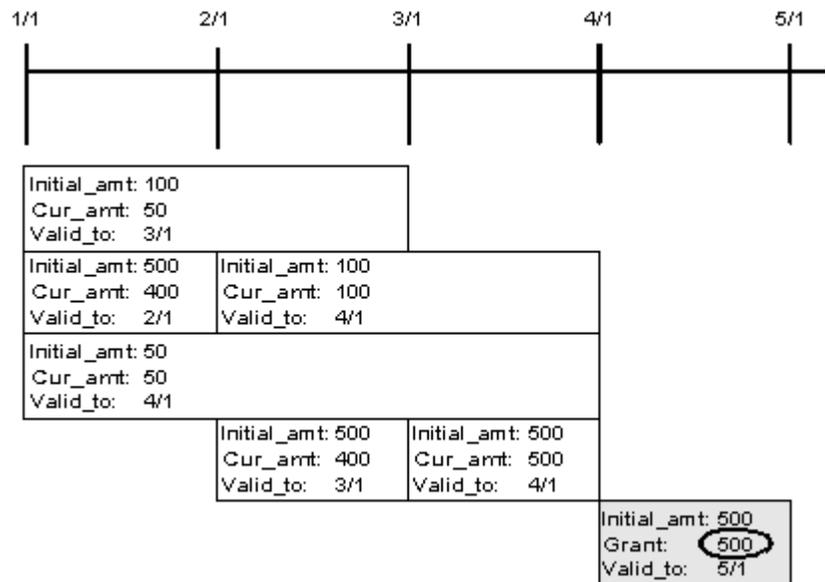
Figure 3–10 March Free Minutes Usage



- On April 1, the cycle forward event creates a new resource sub-balance to track the grant of 500 free minutes for use during April as shown in Figure 3–11.

No free minutes roll over into April from the March and February rollovers because these resources were consumed in March. The 30 free minutes remaining from the January rollover are not rolled over into April because the maximum number of cycles a grant can be rolled over is set to 2.

Figure 3–11 April Free Minutes Grant



The customer has only 500 free minutes available for April.

Important:

- Rollover sub-balances are not truncated or prorated when the corresponding product is canceled.
- If two products contributing to the same balance group have rollover rateplans configured for the same resource with the same rollover frequency, either of the products can be used to roll over the resources. In this case, rollover results may vary depending on the product that is selected first.

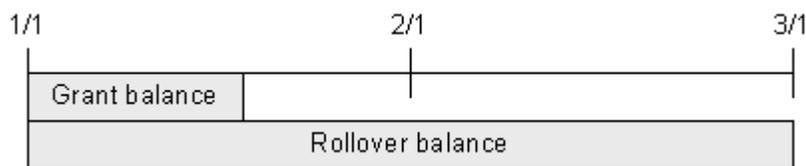
About Rolling Over Resources That Expire in Midcycle

A resource balance can expire in the middle of a cycle: for example, when the resource is valid for only minutes, hours, or days or when the resource is valid for one or more months and starts in the middle of a month.

If a resource’s validity period does not end when the cycle ends, the rollover sub-balance is valid for the entire cycle in which the resource is granted and for the whole of the next cycle.

In [Figure 3–12](#), a monthly cycle event grants 60 free minutes that are valid for two weeks from the grant date. The free minutes are granted on January 1. On January 14, when the balance of free minutes expires, any remaining resources that can be rolled over are added to a new sub-balance, which is valid from January 1 to March 1:

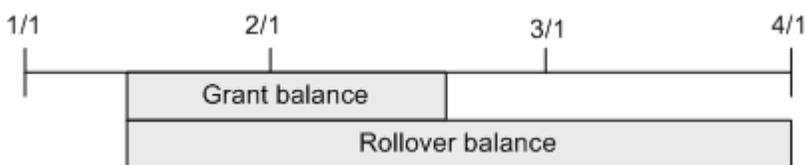
Figure 3–12 Grant and Rollover Balance Date Validity Example 1



The same rollover period applies when the validity period of the free minutes ends after the cycle in which they are granted. In [Figure 3–13](#), a monthly cycle event grants 300 free minutes that are valid for six week from the grant date. The free minutes are granted on January 1. On February 11, when the balance expires, any remaining resources that can be rolled over are added to a new sub-balance, which is valid from January 1 to March 1:

If the resource's validity period does not start at the beginning of the last cycle, the resources are rolled over for one entire cycle.

Figure 3–13 Grant and Rollover Balance Date Validity Example 2



Prorating Rollover Resources

Customers typically purchase and cancel products at some point in the middle of a cycle. When you set up a rollover in Pricing Center, you can specify whether rollover

resources are prorated for the first and last cycles based on the number of days in the cycles that the product is owned.

Table 3–11 describes the rollover proration options:

Table 3–11 Rollover Proration Options

Proration Option	Description
Rollover entire amount	Roll over the available monthly rollover resources.
No rollover	Do not roll over any available monthly rollover resources.
Prorate rollover amount	Calculate the rollover resources based on the percentage of the cycle that the customer owned the rollover.

If you choose to prorate the rollover amount, BRM uses the equation in Figure 3–14 to calculate the amount to rollover:

Figure 3–14 Calculation for Amount to Rollover

$$\frac{\text{ValidFrom} - \text{ValidTo}}{\text{DaysInCycle}} \times \text{Rollover}$$

where:

- ValidFrom is the validity period’s starting date. For example, if the validity period is from March 15 to April 14, ValidFrom is March 15.
- ValidTo is the validity period’s ending date. For example, if the validity period is from March 15 to April 14, ValidTo is April 14.
- DaysInCycle is the number of days in the current cycle. For example, if the validity period is from March 15 to April 14, DaysInCycle is 31.
- Rollover is the available monthly rollover resource, such as 100 minutes.

For example, suppose:

- A customer buys a product on January 15 that grants 500 free minutes of use during each cycle.
- A rollover is set up so that up to 200 unused free minutes roll over into the following cycle.
- The usage cycle is a calendar month.
- The customer does not use any minutes during the first cycle.

Table 3–12 describes how the rollover is handled, depending on the Purchase Mid-cycle Proration setting:

Table 3–12 Impact of Proration Options

Proration Option	Amount Rolled Over into February
Rollover entire amount	200 free minutes are rolled over from January into February. In February, the customer has 700 free minutes (200 rolled over plus the 500 free minutes February grant).

Table 3–12 (Cont.) Impact of Proration Options

Proration Option	Amount Rolled Over into February
No rollover	No free minutes are rolled over from January into February. In February, the customer has only the February grant of 500 free minutes.
Prorate rollover amount	Approximately 110 free minutes are rolled over from January into February: $(17/31) * 200 = 109.67$ In February, the customer has about 610 free minutes (110 rolled over plus the 500 free minutes February grant).

A similar set of results applies to the last-month proration calculation specified in the Cancel Mid-cycle Proration setting.

About Rolling Over Free Resources during Plan Transition

When your customers transition from one plan to another, you can specify that free non-currency resources be rolled over from one plan to another if both plans belong together to at least one plan list.

For more information on plan transitions, see "[Transitioning between Plans](#)".

To specify free resources rollover between plans during plan transition, you must perform these tasks:

- If you use a custom application for customer management, provide the trigger for controlled rollover in the input flist to the PCM_OP_SUBSCRIPTION_TRANSITION_PLAN opcode.
- When you create your plan lists, ensure that plans between which you want to allow rollovers are in at least one plan list together.

For more information, see "[About Plan Lists](#)".

These general rules apply to controlled rollovers:

- Controlled rollovers are not affected by the product cycle rollover settings.
For more information on cycle rollovers, see "[About Rollovers](#)".
- A controlled rollover does not count as a rollover and hence is not restricted to the maximum rollover quantity and the rollover units per period settings for the cycle rollover.
- The cycle grants of free resources that have been prorated during the plan transition are rolled over to the next plan.

Policy-Driven Charging

This chapter describes how to manage policy-driven charging of services by using Oracle Communications Billing and Revenue Management (BRM).

About Policy-Driven Charging in BRM

Policy-driven charging enables you to track a subscriber's service usage and, based on that usage, change the customer's quality of service (QoS) in real-time rating and in offline rating for offline events such as billing and adjustments.

For example, a subscriber purchases a plan for a certain QoS to download video content. The subscriber makes his choice from one of many plans that you have configured with gradations in the QoS based on usage amounts in megabytes, such as 100-150, 150-200, and 200-250 megabytes. When the subscriber starts downloading video content from the network, you can track the quantity of megabytes that the subscriber downloads during that session. When you find that the downloaded quantity crosses the upper threshold set for the selected QoS (for example 150 megabytes), you can use BRM's policy-driven charging to make a seamless change in the policy set for the subscriber's (video downloading) session on the network and allow a change to the QoS from the current to the next level.

About Setting Up Policy-Driven Charging

To set up policy-driven service charging:

- Create offer profiles to define ranges in the QoS for the services you offer, and include the offer profiles in your price lists.

For example, set up high, medium, and low QoS for downloading data from the network, using a noncurrency resource called *Megabytes Used*. Create an offer profile with a unique name (for example, *Platinum*). In the offer profile, specify the resource ID **100009** for the noncurrency resource and set the ranges to 100-150, 150-200, and 200-250 megabytes.

- Link the offer profile to the product or discount with which it is associated so that BRM can track the usage correctly.

For example, connect the offer profile called *Platinum* to a product or discount by using *Platinum* as the provisioning tag for the product or discount. Use the resource ID **100009** for the non-currency resource *Megabytes Used* configured in the offer profile as the non-currency resource in the rate plan for the product or discount.

- Set up provisioning rules that state how the network is to provide and manage the services you offer.

For example, when a subscriber's usage of a video downloading service reaches or crosses a threshold set in the offer profile (for example, 150 megabytes of data download), a different rule can be enforced for the network session, based on the subscriber's profile, his balances, his usage of the resource, and the traffic on the network.

You configure these provisioning rules in the application you use to provide network policy control. Policy controllers (such as Oracle Communications Policy Controller) manage the QoS, optimize high bandwidth traffic, and enforce usage quota levels for subscribers using the network.

The policy controller installs the provisioning policy on the network element responsible for managing the enforcement of network policies on traffic passing through the network.

Note: Setting up provisioning rules to use in policy-driven charging for network capacity and services is beyond the scope of this document. For more information, see the appropriate references.

In addition to configuring offer profiles for provisioning rules in the policy controller, you must configure the following:

- Subscriber preferences for communications, such as whether the subscriber wants to be notified, in what language, and so on.
- Other factors (such as rates) that you use to give better services to more valuable subscribers when network capacity is limited during peak time or in an area.
- A network connectivity application to start, monitor, and end the online charging services for sessions in the network.

Network connectivity applications (such as Oracle Communications Online Mediation Controller) communicate with BRM to manage charging sessions.

Note: Setting up the network connectivity application for policy-driven charging is beyond the scope of this document. For more information, see the appropriate references.

- Event notification to trigger policy-driven charging operations for real-time charging.

How Policy-Driven Charging Works

BRM uses the data you configure in offer profiles to provide you with offer profile threshold breach notifications for both in-session and out-of-session rating events.

Suppose you configure an offer profile called *Platinum* with specific usage ranges for downloading video content. The non-currency resource such as *Megabytes (MB) Used* and its usage ranges such as 100-150 megabytes, at 150-200 megabytes, over 200-250 megabytes are defined within a policy label (called *Fair Usage*).

When your subscriber initiates the downloading of a video,

- The policy controller communicates with BRM to set up a provisioning policy for the session.

A provisioning rule is established on the network for the session for an initial quantity, for example, 35 megabytes. The range for the current QoS for the subscriber is 100-150 megabytes.

- The download begins with the network connectivity application managing the online-charging session.

Throughout that session, whenever the subscriber requests more quota, BRM calculates whether it can allocate the requested amount or if it must readjust the quota.

For example, BRM receives a request with the information that the subscriber has used 35 megabytes and needs 75 megabytes more. BRM does the following:

- Calculates how much the subscriber has used.

BRM uses the balance information and consumed reserved amount (across parallel sessions, iPhone, video, and computer) for that subscriber. For example, BRM calculates that the subscriber has used a total of 150 megabytes. With 100-150 megabytes as the range for the current QoS for the subscriber, his usage is at the threshold for the policy.

- Reduces the allocation, if necessary.

In the example case, the subscriber has used a total of 150 megabytes. The upper thresholds of the offer profiles are set at 150, 200, and 250 megabytes. The next threshold is 200 megabytes. The available quota for that QoS is 50 megabytes only (200-150). The subscriber has requested 75 megabytes, an amount which is greater than the available quota of 50 megabytes.

In this scenario, BRM readjusts the quota allocation to 50 megabytes.

When the used amount for the resource crosses a set threshold (150, 200, 250, and so on), the provisioning rule (which was tied to the previous policy level) can no longer be applied for the session.

BRM and the policy controller ensure that a more appropriate rule is established for the session on the network in the following way:

- BRM sends an offer profile threshold breach notification to the policy controller with information on the threshold breach (150 in our example case) and information on the subscriber's preferences for receiving communications.
- The policy controller does the following:
 - Communicates with BRM to retrieve the latest information on the subscriber and his usage of the service.
 - Arrives at the new rule which enables the increase in bandwidth.
 - Establishes that provisioning rule for the session.

About Notifications Related to Offer Profiles Sent by BRM

When you set up policy-driven charging for your services, BRM sets up notification events for the following:

- When you create, modify, or delete the following:
 - Offer profiles
 - Products associated with offer profiles
 - Discounts associated with offer profiles
- Offer profile threshold breaches

Offer profile threshold breaches can occur with your subscriber's usage of a resource in a prepaid session. Or, they can occur when BRM processes offline events, such as billing and adjustments.

BRM places these offer profile threshold breach notifications in a central Oracle Advanced Queuing (AQ) database queue for use by customer relationship management (CRM) applications. You can use these applications to retrieve data directly from the Oracle AQ database queue and use the information in these offer profile threshold breach notifications. See *BRM Developer's Guide* for more information on the Oracle AQ database queue.

BRM provides the following information in offer profile threshold breach notifications associated with policy-driven charging:

- Service identifier
- List of aliases that also identify this user/service (ALIAS_LIST)
- Name of the resource
- Resource ID
- Status label from the offer profile

See "[Sample Threshold Breach Notification](#)".

Configuring Your BRM Environment to Support Policy-Driven Charging of Services

Configure your BRM environment to support policy-driven charging of services by completing the following:

1. Set up offer profiles in your price lists. See "[About Setting Up and Managing Offer Profiles](#)".
2. Configure BRM to track a subscriber's usage of the resources you offer. Set up BRM to create notifications tailored to the subscriber's preferences on how they want to be notified when their usage exceeds the threshold set in their offer profiles. See "[About Configuring Resources for Tracking by BRM](#)".
3. Merge the event notification entries specific to policy-driven charging with the BRM event notification list. See "[Configuring Event Notification for Policy-Driven Charging](#)".
4. Enable your enterprise applications to integrate with policy-driven charging in BRM. See "[Enabling Enterprise Applications to Handle Policy-Driven Charging Events](#)".
5. Update the BRM database with reservation preferences configuration for policy-driven charging. See "[Updating Reservation Preferences Configuration for Policy-Driven Charging](#)".
6. Ensure that the Connection Manager can forward the notifications associated with policy-driven charging. See "[Updating the Connection Manager for Policy-Driven Charging](#)".

About Setting Up and Managing Offer Profiles

You can create offer profiles when you create a price list or add your offer profile data to an existing price list. See "[Setting Up Offer Profiles](#)" for information on setting up the data for offer profiles.

About Using New Resource IDs in Your Offer Profiles

When you start (or restart) your BRM system, BRM caches the set of unique resource IDs that you have configured in your offer profiles. BRM processes a subscriber's usage only if the resource ID is present in its cache.

Note: When you introduce a resource ID that has not been used in BRM, you must update the entries in this cache. To do so, stop and start the Connection Manager (CM).

Configuring Offer Profile Data

BRM provides sample offer profile data in the default `pin_offer_profile_policy_labels.xml` file.

This sample file uses the format detailed in the `BRM_Home/PricingCenter/price_list.xsd` file. `BRM_Home` is the directory in which you installed the BRM software. Maintain your offer profiles in the format used in the sample file. For more information on using the XML interface, see "[Using the XML Pricing Interface to Create a Price List](#)".

Important:

- Offer profile names must be unique.
 - If you introduce a new resource ID in an offer profile you add to or modify in the database, you must restart the CM after your load the offer profiles.
-
-

To configure your offer profile data:

1. Go to the `BRM_Home/setup/scripts` directory.
2. Open the `pin_offer_profile_policy_labels.xml` file in an XML editor or text editor.

Tip: Save a copy of the default file before you make any changes to it.

3. Edit this file using the parameters as shown in the file. [Table 4-1](#) describes the elements.

Table 4-1 Elements Used to Store Offer Profiles

Element	Description
label	Name of the policy label
offer_profile	Array to hold offer profiles
offer_profile_code	Key identifier associated with an offer profile
offer_profile_name	Name of an offer profile
policy_label	Array to hold policy labels
price_list	Price List object within which the offer profile array resides
resource_id	Id used for the resource associated with a policy label
resource_name	Name of the resource associated with a policy label
status_label	Status label name indicating the level of the tier

Table 4–1 (Cont.) Elements Used to Store Offer Profiles

Element	Description
tier	The tier as an array with each entry made up of a status label, and the start and end of the range for that tier level.
tier_end_range	Start of the range for a tier level
tier_start_range	End of the range for a tier level
unit	Unit of the resource associated with a policy label. The values are: <ul style="list-style-type: none"> ▪ None (0) ▪ Second (2) ▪ Minute (1) ▪ Hour (3) ▪ Day (4) ▪ Byte (11) ▪ Kilobyte (12) ▪ Megabyte (13) ▪ Gigabyte (14) The default value is 0.

[Example 3–1, "Sample Policy Label"](#) shows a sample offer profile.

4. Save the `pin_offer_profile_policy_labels.xml` file.

You are now ready to store the offer profiles in the BRM database.

Managing Offer Profile Data

BRM stores offer profiles in `/offer_profile` objects. Use the `loadpricelist` utility to load, retrieve, modify, and delete `/offer_profile` objects.

Note: Before you use this utility, verify that:

- The XML offer profile file is complete and follows the guidelines.
- The `Infranet.properties` file in your CLASSPATH has the correct login information for the BRM database to which you want to connect.

For more information, see ["Prerequisites for the XML Pricing Interface"](#).

Storing Offer Profile Data

Important: If you introduce a new resource ID in an offer profile you add to the database, you must restart the CM after you load the offer profiles.

To do so, stop and restart the CM.

Use the following command to load offer profiles into your BRM database:

```
loadpricelist -f -v -c pin_offer_profile_policy_labels.xml
```

where:

- **-f** executes the command without prompting for a confirmation.
- **-v** displays information about successful or failed processing as the utility runs
- **-c** reads the offer price information from **pin_offer_profile_policy_labels.xml** and commits it to the BRM database.

The offer profile information is now stored in **/offer_profile** objects in the database.

Loading Modified Offer Profiles in the Interactive Mode

You can commit the modified offer profile data by using the **loadpricelist** utility when you are working in the interactive mode.

To load offer profile data only:

```
write PriceListFile offer_profile PriceObject
```

where:

- *PriceObject* is the specified pricing object in the database.
- *PriceListFile* is the entire price list in the database.

For more information on the utility, see "[loadpricelist](#)".

Retrieving Offer Profiles

To retrieve offer profile data, use the **loadpricelist** utility in the following ways:

- Non-Interactive mode:


```
loadpricelist -r PriceListFile
```
- Interactive mode:


```
read PriceListFile
```

The complete price list is returned in the *PriceListFile* XML file. Use your custom code to retrieve the **offer_price** objects contained in that price list.

Modifying Offer Profiles

Important: If you introduce a new resource ID in an offer profile you modify, you must restart the CM after you load the offer profiles.

To modify the offer profile data in the BRM database:

1. Retrieve the offer profile data and use your custom code to retrieve the **offer_price** objects. See "[Retrieving Offer Profiles](#)".
2. Modify the **offer_price** objects as necessary.
3. Configure the offer profile data in XML format. See "[Configuring Offer Profile Data](#)".
4. Commit the modified offer profile data into the database. See "[Storing Offer Profile Data](#)".
5. If you added a new resource ID to any offer profile, stop and restart the CM.

Deleting Offer Profiles

Delete offer profiles using the `loadpricelist` utility in the following ways:

- Non-Interactive mode:

```
loadpricelist -p
```

This command deletes all the pricing objects from the database.

- Interactive mode:

```
delete offer_profile PriceObject
```

Note: If you use the `delete` command without any parameters, all in-memory pricing information is deleted from the database.

About Configuring Resources for Tracking by BRM

Configure the resources that BRM should track by setting up provisioning tags for your services as shown in the `BRM_Home/sys/data/config pin_offer_profile_provisioning_tags_policy_attributes.xml` file. For each provisioning tag, configure the types of resources valid for that tag and provide information on the specific opcode which must be called to process changes to the product or discount associated with that provisioning tag.

The `pin_offer_profile_provisioning_tags_policy_attributes.xml` file contains the configuration for the default provisioning tag which BRM provides. See [Example 5-1, "Default Provisioning Tag Configuration"](#). The first section of this file lists the valid resources for the default provisioning tag. The next section contains information on the opcodes to be called when the product or discount containing the provisioning tag is purchased or canceled, the parameters that specify the fields to be added to the opcode's input list, and the value for each field.

See "[Configuring Provisioning Tags for Policy-Driven Charging](#)" for more information.

Configuring Event Notification for Policy-Driven Charging

To configure event notification for policy-driven charging, you must merge the policy-driven charging event notification list with the general BRM event notification list.

Each event in an event notification list is mapped to the opcode or opcodes that are executed when the event occurs. The event notification list for policy-driven charging contains entries for offer profiles and for the products and discounts associated with them.

[Example 4-1](#) shows the event notification list for offer profiles:

Example 4-1 Event Notifications Related to Offer Profiles

```
1301 0 /event/notification/offer_profile/create
1301 0 /event/notification/offer_profile/update
1301 0 /event/notification/offer_profile/delete
1301 0 /event/notification/offer_profile/ThresholdBreach
```

BRM uses the Enterprise Application Integration (EAI) framework publishing opcode, `PCM_OP_PUBLISH_GEN_PAYLOAD` (number 1301), to set up event notifications for offer profiles. This opcode is internal to BRM.

[Example 4-2](#) shows the event notification list for products and discounts associated with offer profiles:

Example 4-2 Billing and Product Event Notification Entries

```
301    0    /event/billing/product/action/purchase
301    0    /event/billing/product/action/modify
301    0    /event/billing/product/action/cancel
301    0    /event/billing/product/action/modify/status
301    0    /event/billing/discount/action/purchase
301    0    /event/billing/discount/action/modify
301    0    /event/billing/discount/action/cancel
301    0    /event/billing/discount/action/modify/status
```

The **301** entry is the opcode number for the PCM_OP_ACT_POL_EVENT_NOTIFY policy opcode. See ["About the PCM_OP_ACT_POL_EVENT_NOTIFY Policy Opcode"](#).

For more information on event notification, see "Using Event Notification" in *BRM Developer's Guide*.

Merging the Policy-Driven Event Notification List with the BRM Event Notification List

To merge the policy-driven event notification list with the BRM event notification list:

1. Verify that the list entries for policy-driven charging are enabled. See ["Verifying That Policy-Driven Charging Event Notification List Entries Are Enabled"](#).
2. Merge the list for policy-driven charging with the general BRM list. See ["Merging Event Notification Lists"](#).
3. Load the updated event notification data into the BRM database. See ["Loading the Updated Event Notification File into the BRM Database"](#).

Verifying That Policy-Driven Charging Event Notification List Entries Are Enabled

To verify that the policy-driven charging event notification list entries are enabled:

1. Open the `BRM_Home/sys/data/config/pin_notify_ipc` file in a text editor.

Tip: Save a copy of the default file before you change it.
2. Verify the following:
 - The offer profile notifications are enabled as shown in [Example 4-1](#).
 - The billing event notifications for product and discount objects are enabled as shown in [Example 4-2](#).
3. Save and close the file.

For more information, see "Editing the Event Notification List" in *BRM Developer's Guide*.

Merging Event Notification Lists

Merge the event notification list in the `pin_notify_ipc` file with the event notification list in the `pin_notify` file. For more information, see "Merging Event Notification Lists" in *BRM Developer's Guide*.

Loading the Updated Event Notification File into the BRM Database

To enable event notification for policy-driven charging, run the `load_pin_notify` utility to load the merged event notification list into the BRM database. For more information, see "Loading the Event Notification List" in *BRM Developer's Guide*.

Enabling Enterprise Applications to Handle Policy-Driven Charging Events

Complete the following operations to enable your enterprise applications to handle policy-driven charging event notifications that BRM publishes:

1. [Verifying That BRM Is Integrated with Your Enterprise Applications](#)
2. [Enabling Enterprise Applications to Process Policy-Driven Charging Events](#)
3. [Specifying the Payload Configuration File to Use](#)

For more information, see "Integrating BRM with Enterprise Applications" in *BRM Developer's Guide*.

Verifying That BRM Is Integrated with Your Enterprise Applications

For your enterprise applications to use the offer profile event notifications published by BRM, ensure that BRM is integrated with other applications in your enterprise. To do so, ensure that EAI Manager is installed.

For more information, see "About Installing EAI Manager" in *BRM Developer's Guide*.

Enabling Enterprise Applications to Process Policy-Driven Charging Events

You enable your enterprise applications to receive the notifications that BRM publishes when it processes policy-driven charging events.

When a customer buys, modifies, cancels a product or discount that is associated with an offer profile, BRM publishes a business event (such as **ProductPurchase** when the customer buys a product). If an offer profile is created, modified, deleted or there is an threshold breach, BRM publishes a related business event (for example, **OfferProfileCreate**, when an offer profile is initially stored in the database).

Business events associated with policy-driven charging are defined in the *BRM_Home/sys/eai_js/payloadconfig_ipc.xml* payload configuration file. BRM provides default definitions of business events in the *BRM_Home/sys/eai_js/payloadconfig.xml* payload configuration file.

If you already use a payload configuration file (for example, `payloadconfig.xml`) in your BRM environment, set up one payload configuration file to contain the definitions of all your business events.

Setting Up One Payload Configuration File

Set up one payload configuration file by doing the following:

1. Merge the contents of `payloadconfig_ipc.xml` (the payload configuration file for policy-driven charging) with your current payload configuration file (such as, `payloadconfig.xml`).

Tip: Save a copy of the default payload configuration files before merging them.

2. Save the merged file under a different name (for example, *your_payloadconfig.xml*).

Specifying the Payload Configuration File to Use

To specify the payload configuration file to use:

1. Open *BRM_Home/sys/eai_js/Infranet.properties* in a text editor.
This is the payload generator external module **eai_js** properties file.
2. Point the **infranet.eai.configFile** entry to the appropriate payload configuration file.

For example:

```
infranet.eai.configFile=/pinhome/pin201/opt/portal/7.5/sys/eai_js/your_
payloadconfig.xml
```

where *your_payloadconfig.xml* is the merged payload configuration file containing the business event entries associated with policy-driven charging.

3. Save and close the file.
4. Restart **eai_js**.

For more information, see "Configuring the EAI payload for the Synchronization Queue DM" in *BRM Synchronization Queue Manager*.

Updating Reservation Preferences Configuration for Policy-Driven Charging

To update the reservation preferences configuration for services associated with policy-driven charging:

1. Add information on how the resource is to be rated (for example, by duration, by volume, by both duration and volume, or by occurrence). See "[Updating Reservation Preferences Configuration Settings for Resources](#)".

Note: Policy-driven charging of services does not support prerated events.

2. Load the reservation preferences into the BRM database. See "[Loading Reservation Preferences for Policy-Driven Charging](#)".

Updating Reservation Preferences Configuration Settings for Resources

BRM stores the default entries for reservation preferences associated with policy-driven charging in the **pin_config_reservation_aaa_prefs_XXX** file. For example:

- **pin_config_reservation_aaa_prefs_gsm_telephony** (GSM telephony)
- **pin_config_reservation_aaa_prefs_gsm_data** (GSM data)
- **pin_config_reservation_aaa_prefs_gprs** (GPRS)

To configure the resource ID for the appropriate counters:

1. Open the *BRM_Home/sys/data/config/pin_config_reservation_aaa_prefs_XXX* file in a text editor.

For our example, open the `BRM_Home/sys/data/config/pin_config_reservation_aaa_prefs_gsm_data` file, where `BRM_Home` is the directory in which BRM is installed.

2. Add an entry to specify the appropriate resource ID with the entity to be rated, such as duration (2), volume (3), duration and volume (4), or occurrence (8).

For example:

```
1 PIN_FLD_RESOURCE_ID INT [0] 1000009 in REQ_MODE 4
```

Here, a noncurrency resource, (Megabytes Used) with the resource ID **100009** is associated with a volume-based request (**REQ_MODE 4**).

3. Save and close the file.

For more information, see "Editing the Event Notification List" in *BRM Developer's Guide*.

Loading Reservation Preferences for Policy-Driven Charging

To load the reservation preferences list for policy-driven charging:

1. Go to the `BRM_Home/sys/data/config` directory.
2. Use the following command to run the `load_config_reservation_aaa_prefs` utility:

```
load_config_reservation_aaa_prefs -d -v load_config_reservation_aaa_prefs_XXX
```

where:

- `-d` creates a log file for debugging purposes.
- `-v` displays information about successful or failed processing as the utility runs.
- `load_config_reservation_aaa_prefs_XXX` is the reservation configuration file.

For example:

```
load_config_reservation_aaa_prefs -d -v pin_config_reservation_aaa_prefs_gsm_data
```

For information on the `load_config_reservation_aaa_prefs` utility, see "load_config_reservation_aaa_prefs" in *BRM Telco Integration*.

3. To verify that the reservation preferences were loaded, display the `/config/reserve` object by using Object Browser or the `robj` command with the `testnap` utility.
4. Stop and restart the CM.

For more information, see "Specifying Default Authorization and Reauthorization Values" in *BRM Telco Integration*.

Updating the Connection Manager for Policy-Driven Charging

If you use offer profiles in price lists, ensure that you update the CM configuration file, `BRM_Home/sys/cm/pin.conf`, for the following:

- Configure the memory required for offer profile cache. See ["About Configuring the Required Memory for Offer Profiles Cache"](#).
- Reference the system environment variables in the `pin.conf` file. See ["About Referencing Offer Profile Related Variables in pin.conf Files"](#).

About Configuring the Required Memory for Offer Profiles Cache

The offer profile cache called `fm_offer_profile_cache` is loaded with all `/offer_profile` objects at the initialization of the `fm_offer_profile` library. The `fm_offer_profile_cache` cache stores the last updated timestamp along with each `/offer_profile` object.

Important: Restart the CM after you make any change to the offer profile configuration entry in the `pin.conf` file.

To update the `pin.conf` file for the `fm_offer_profile_cache` entry, see ["Updating the pin.conf Configuration File for Offer Profiles"](#).

About Referencing Offer Profile Related Variables in pin.conf Files

You can reference the system environment variable for offer profiles in `pin.conf` configuration file to facilitate future migration of the `pin.conf` file to BRM implementations on other platforms. See "System Administration pin.conf Reference" in *BRM System Administrator's Guide*.

Important: Restart the CM after you make any change to the `/offer_profile` object.

To update the `pin.conf` file for this entry, see ["Updating the pin.conf Configuration File for Offer Profiles"](#).

Updating the pin.conf Configuration File for Offer Profiles

To update the CM `pin.conf` file:

1. Open the `BRM_Home/sys/cm/pin.conf` file in a text editor.)
2. Configure the required memory for the offer profile cache using appropriate values for `number_of_entries`, `cache_size`, `hash_size` in the following command. Add this line, if necessary.

```
- cm_cache fm_offer_profile_cache number_of_entries cache_size hash_size
```

For example:

```
- cm_cache fm_offer_profile_cache 20, 102400, 13
```

For more information, see "Improving Performance for Loading Large Price Lists" in *BRM System Administrator's Guide*

3. Verify that the following command to reference the offer profile environmental variable from within the `pin.conf` file is present in the file:

```
- cm fm_module ${PIN_HOME}/lib/fm_offer_profile${LIBRARYEXTENSION} fm_offer_profile_config fm_offer_profile_init pin
```

Uncomment this command if it is commented, or add this command if it is not present in the `pin.conf` file.

For more information, see "Preparing for Platform Migration by Using Variables in pin.conf Files" in *BRM System Administrator's Guide*.

4. Save the file.

5. Stop and restart the CM.

For more information, see "Improving Connection Manager Performance" in *BRM System Administrator's Guide*.

Customizing Information Received from BRM

BRM uses the following opcodes to support offer profile threshold notifications:

- PCM_OP_BAL_APPLY_MULTI_BAL_IMPACTS
- PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS
- PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD
- PCM_OP_TCF_AAA_POL_GET_SUBSCRIBER_PROFILE
- PCM_OP_ACT_POL_EVENT_NOTIFY

This section describes these opcodes and how you can customize them, if necessary.

About the PCM_OP_BAL_APPLY_MULTI_BAL_IMPACTS Opcode

BRM uses the PCM_OP_BAL_APPLY_MULTI_BAL_IMPACTS opcode to process both offline and online charging events. This opcode retrieves the current balance and calls PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS policy opcode.

About the PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS Policy Opcode

The PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS policy opcode sets up an offer profile threshold breach notification when it encounters a threshold breach in a customer's usage of a resource defined in an offer profile (associated with the customer's purchased rate plan).

When the policy opcode is called by BRM as part of an authorization or reauthorization of a subscriber's request to use a service, it sets up an in-session notification for the offer profile threshold breach it encounters for the customer's usage of the resource. When the policy opcode is called by BRM during billing or accounts receivable operations, it sets up an out-of-session notification for the offer profile threshold breach it encounters for the customer's usage of the resource.

The PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS policy opcode does not handle prerated events.

When you start (or restart) Connection Manager, BRM caches the set of unique resource IDs that you have configured in your offer profiles. The PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS policy opcode uses this cache of resource IDs as the initial filter when applying the balance impact of each resource to retrieve the offer profile information and determine if an offer profile threshold notification is necessary. If a resource is not present in the cached entries, this policy opcode does not retrieve the offer profile information for that resource. Consequently, it will not set up an offer profile threshold breach notification, even if such a notification is necessary.

Important: If you introduce a new resource ID, stop and restart the CM.

How Balances Are Processed for Out-of-Session Notifications

For out-of-session rating events, the PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS policy opcode uses the offer profile associated with the product or discount that granted this resource (called the grantor object).

The policy opcode retrieves all the services associated with the balance group and records an offer profile threshold breach notification event for all the services where an offer profile threshold was breached.

How Balances Are Processed for Notifications for Prepaid Sessions

For in-session rating events, the PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS policy opcode sets up offer profile threshold breach notifications for the subscriber only. It does not take resource sharing into consideration when it sets up these notifications.

This policy retrieves the offer profile associated with the purchased product or discount in the balance impact data it received from the rating process. It uses the offer profile information, the subscriber's current balances and the consumed reservation amount for the resource to determine whether an offer profile threshold was breached. If so, it records an offer profile threshold breach notification event.

Customizing Information in Offer Profile Threshold Breach Notifications

See ["Sample Threshold Breach Notification"](#) for the default information in an offer profile threshold breach notification. You can customize the PCM_OP_BAL_POL_APPLY_MULTI_BAL_IMPACTS policy opcode to retrieve other information on the resource usage.

About the PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD Opcode

The PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD opcode uses offer profiles passed in the input flist to retrieve the dynamic information for a given service identifier and resource ID.

BRM calls this opcode to perform the following operations:

- [Determining Whether Balance Impacts Trigger Notifications](#)
- [Readjusting Quotas for Requests to Update and Authorize](#)
- [Readjusting Quotas for Other AAA Requests](#)

For more information on the PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD opcode, see *Opcode Flist Reference*.

Determining Whether Balance Impacts Trigger Notifications

To determine whether any threshold breach notifications result from balance impacts due to the resource usages, BRM calls the PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD opcode with the balances (without current impact) stored in the BALANCES array of opcode's input flist.

The PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD opcode does the following:

1. Calculates the sum of the current balance and the consumed reservation from the BALANCES array.
2. Locates old used quota and the last threshold notification.

3. Calls the `PCM_OP_BAL_GET_PREPAID_BALANCES` opcode with the resource ID to compute the new used amount.
The `PCM_OP_BAL_GET_PREPAID_BALANCES` opcode returns the sum of the updated current balance and consumed reservation as the new used amount.
4. Locates the status label, offer profile name and policy label name for the tier with the nearest threshold. To do so, it employs the new used amount and the range from the last threshold notification.
5. Calculates the offset to the next threshold as the difference between the new used amount and the next threshold on that tier.
6. Sets the `PIN_FLD_INDICATOR` field in the output to 0 if there is a breach. And returns the calculated offset to the next threshold.

Readjusting Quotas for Requests to Update and Authorize

BRM calls the `PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD` opcode with the additional quota reported by the network in the `QUANTITY` field of the input list.

The `PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD` opcode does the following:

1. Calls the `PCM_OP_BAL_GET_PREPAID_BALANCES` opcode with the resource ID to retrieve the current balance and consumed reservation.
2. Calculates the total used as the sum of the current balance, the consumed reservation, and the additional quantity reported by the network.
3. Locates the status label, offer profile name, and policy label name for the tier with the nearest threshold with reference to the total used amount.
4. Calculates the offset to the next threshold as the difference between the new used amount and the next threshold on that tier.
5. Returns the calculated offset to the next threshold

Readjusting Quotas for Other AAA Requests

When called to readjust the quota for AAA requests other than the update and reauthorize request, the `PCM_OP_OFFER_PROFILE_CHECK_POLICY_THRESHOLD` opcode does the following:

1. Retrieves the current balance and consumed reservation by calling the `PCM_OP_BAL_GET_PREPAID_BALANCES` opcode with the resource ID.
2. Calculates the total used as the sum of the current balance and the consumed reservation.
3. Locates the status label, offer profile name and policy label name for the tier with the nearest threshold with reference to the total used amount.
4. Calculates the offset to the next threshold as the difference between the new used amount and the next threshold on that tier.
5. Returns the calculated offset to the next threshold.

About the `PCM_OP_TCF_AAA_GET_SUBSCRIBER_PROFILE` Opcode

The `PCM_OP_TCF_AAA_GET_SUBSCRIBER_PROFILE` opcode returns the following information from the database:

- All the information associated with a subscriber's mobile station ID (MSID)
- Specific service only for a given object identifier (POID)

The `PCM_OP_TCF_AAA_GET_SUBSCRIBER_PROFILE` opcode uses the internal `PCM_OP_OFFER_PROFILE_GET_OFFER_PROFILE` opcode to fetch the necessary information. It returns the information in the `PIN_FLD_RESULTS` array with `PIN_FLD_ACCOUNT` containing information associated with the account and `PIN_FLD_SERVICE_INFO` containing information on the services.

The opcode returns the following information based on the value you input for the `PIN_FLD_MODE` entry when you call this opcode:

- Static information on the resource used by a subscriber such as offer profiles and subscriber's preference data (`PIN_FLD_MODE` is 1)
- Dynamic information such as current balances for the various resources (`PIN_FLD_MODE` is 2)
- Both types of information (`PIN_FLD_MODE` is 3)

For more information on the `PCM_OP_TCF_AAA_GET_SUBSCRIBER_PROFILE` opcode, see *Opcode List Reference*.

Customizing Information Received from BRM

Use the `PCM_OP_TCF_AAA_POL_GET_SUBSCRIBER_PROFILE` policy opcode to customize the information your network policy controlling application retrieves from BRM.

By default, `PCM_OP_TCF_AAA_POL_GET_SUBSCRIBER_PROFILE` policy opcode calls the `PCM_OP_TCF_AAA_GET_SUBSCRIBER_PROFILE` opcode and returns the output from that opcode.

For more information on the `PCM_OP_TCF_AAA_POL_GET_SUBSCRIBER_PROFILE` policy opcode, see *Opcode List Reference*.

About the `PCM_OP_ACT_POL_EVENT_NOTIFY` Policy Opcode

When a product or discount associated with an **offer profile** is purchased, canceled, or modified, the `PCM_OP_ACT_POL_EVENT_NOTIFY` policy opcode calls the `PCM_OP_OFFER_PROFILE_GET_OFFER_PROFILE` opcode to retrieve the associated offer profile data and adds it to the event notification. The `PCM_OP_OFFER_PROFILE_GET_OFFER_PROFILE` opcode is internal to BRM. See "[About the `PCM_OP_TCF_AAA_GET_SUBSCRIBER_PROFILE` Opcode](#)" for more information.

For more information on the `PCM_OP_ACT_POL_EVENT_NOTIFY` policy opcode, see *Opcode List Reference*.

Customizing Information in Event Notifications for Policy-Driven Charging

Use the `PCM_OP_ACT_POL_EVENT_NOTIFY` policy opcode to customize the event notification information associated with policy-driven charging that you want to receive from BRM.

Managing Offer Profiles with Opcodes

You can manage offer profile data using price list opcodes and offer profile opcodes.

Managing Offer Profile Data with Price List Opcodes

Use the PCM_OP_PRICE_SET_PRICE_LIST and PCM_OP_PRICE_GET_PRICE_LIST opcodes to manage your offer profiles.

Creating, Modifying, and Deleting Offer Profiles with PCM_OP_PRICE_SET_PRICE_LIST

The PCM_OP_PRICE_SET_PRICE_LIST opcode searches the database for each offer profile in the PIN_FLD_OFFER_PROFILES array. To create or modify offer profiles in the database, input the **/offer_profile** objects in the PIN_FLD_OFFER_PROFILES array. To delete an offer profile from the database, input the **/offer_profile** object in the PIN_FLD_OFFER_PROFILES array and specify the PIN_FLD_DELETE_FLAG entry in the input flist.

The PCM_OP_PRICE_SET_PRICE_LIST opcode searches the database for each offer profile in the PIN_FLD_OFFER_PROFILES array and does one of the following:

- If the offer profile is not found, the opcode creates and stores an **/offer_profile** object with this data.
- If the offer profile is found, the opcode retrieves that **/offer_profile** object and does one of the following:
 - If the PIN_FLD_DELETE_FLAG entry is not present in the input flist, the opcode updates **/offer_profile**, and stores the updated object.
 - If the PIN_FLD_DELETE_FLAG entry is present in the input flist, the opcode deletes the **/offer_profile** object from the database.

See "[Managing Price Lists](#)" for more information.

Retrieving Offer Profiles with PCM_OP_PRICE_GET_PRICE_LIST

Use the PCM_OP_PRICE_GET_PRICE_LIST opcode to retrieve the contents of an **/offer_profile** object from the database. The PIN_FLD_OFFER_PROFILES array in the output contains the **/offer_profile** objects.

See "[Managing Price Lists](#)" for more information.

Managing Offer Profiles with Offer Profile Opcodes

Manage offer profile data using the following:

- PCM_OP_OFFER_PROFILE_SET_OFFER_PROFILE. See "[About the PCM_OP_OFFER_PROFILE_SET_OFFER_PROFILE Opcode](#)".
- PCM_OP_TCF_AAA_POL_GET_SUBSCRIBER_PROFILE. See "[About the PCM_OP_TCF_AAA_GET_SUBSCRIBER_PROFILE Opcode](#)".

About the PCM_OP_OFFER_PROFILE_SET_OFFER_PROFILE Opcode

Use the PCM_OP_OFFER_PROFILE_SET_OFFER_PROFILE opcode to create, modify, and delete offer profiles.

Input the offer profiles in the PIN_FLD_OFFER_PROFILES array. To delete an offer profile, specify the PIN_FLD_DELETE_FLAG entry for the offer profile in the input flist.

The opcode searches the database for the offer profiles in the PIN_FLD_OFFER_PROFILES array and does one of the following:

- If the offer profile is not found, the opcode creates and stores an **/offer_profile** object with this data.
- If the offer profile is found, the opcode retrieves that **/offer_profile** object and
 - If the PIN_FLD_DELETE_FLAG entry is not present in the input flist, it updates **/offer_profile**, and stores the updated object.
 - If the PIN_FLD_DELETE_FLAG entry is present in the input flist, it deletes the **/offer_profile** object.

For more information on the PCM_OP_OFFER_PROFILE_SET_OFFER_PROFILE opcode, see *BRM Developer's Reference*.

Customizing Subscriber Preferences Data for Policy-Driven Charging

BRM stores the default entries for subscriber preferences associated with policy-driven charging in the **config_subscriber_preferences_map.xml** file.

To customize subscriber preferences data for policy-driven charging:

1. Open the *BRM_Home/sys/data/config/config_subscriber_preferences_map.xml* configuration file in a text editor.
2. Edit the file as necessary. You can add other subscriber preferences.

The elements in the XML file are described in the discussion on subscriber preferences in *BRM Telco Integration*.

3. Save and close the file.

See "Managing Subscriber Preferences" in *BRM Telco Integration*.

Customizing Business Events Associated with Policy-Driven Charging

If you customize business events associated with policy-driven charging, include your definitions of these events in the **payloadconfig_ipc.xml** configuration file. For more information, see "About Publishing Additional Business Events" in *BRM Developer's Guide*.

Sample Threshold Breach Notification

[Example 4-3](#) shows the contents of a sample offer profile threshold breach notification.

Example 4-3 Sample Offer Profile Threshold Breach Notification

```
PIN_EVENT_TY('ThresholdBreach', '<?xml version="1.0"?>
<ThresholdBreach Version="1.0">
  <service_obj>0.0.0.1 /service/telco 223789 7</service_obj>
  <account_obj>0.0.0.1 /account 10 1 </account_obj>
  <login>test</login>
  <alias_list>
    <name>694-20120607-231907-3-9178 --157566752-slc.example.com</name>
  </alias_list>
  <resource_list>
    <resource_name>Megabytes Used</resource_name>
    <resource_id>100009</resource_id>
    <current_bal>35.28</current_bal>
    <unit>0</unit>
    <threshold_breach >
  </threshold_breach >
  <status_label>HIGH_QOS</status_label>
```

```
<delta_to_next_threshold>14.72</delta_to_next_threshold>
<policy_label>Fair Usage</policy_label>
<offer_profile_name>platinum</offer_profile_name>
</threshold_breach>
</resource_list>
</ThresholdBreach>
```

Working with Provisioning Tags

This chapter describes how to implement Oracle Communications Billing and Revenue Management (BRM) provisioning tags.

About Provisioning Tags

Provisioning tags implement extended rating attributes (ERAs) or other user-defined attributes that can enable rating or discounting to vary for a service. For telco services, you can also use provisioning tags to provision supplementary services and service extensions.

An offer profile and the provisioning tag for the associated product or discount use the same name and that name must be unique. If you create an offer profile to associate with an existing product or discount, use the provisioning tag to name the offer profile. If you configure a new product or discount around an existing offer profile, use the appropriate offer profile name as the provisioning tag for the product or discount.

You can use one of the following methods to define provisioning tags, depending on what the tag is for:

- **Provisioning tag framework:** You create provisioning tags by defining them in an XML file that is loaded into a configuration object in BRM. You can create provisioning tags for any service or for an account.

You also might need to add the tag to the provisioning policy source file and compile the file.

See "[Using the Provisioning Tag Framework](#)".

Important:

- For telco services, use Services Framework Manager to define provisioning tags in most cases. You must use Services Framework Manager if the provisioning tag creates supplementary services or service extensions. See "About Provisioning Tags for Telco Services" in *BRM Telco Integration*.
- For non-telco services, the provisioning tag framework is the recommended method of creating provisioning tags.

-
-
- **Provisioning Tags application in Pricing Center:** For telco services only, you can create service-level provisioning tags with this application. You can include service-level ERAs, supplementary services, and service extensions in the tags.

Important: You can create provisioning tags that include existing ERAs only.

See "About the Provisioning Tags Application" in *BRM Telco Integration*.

- **Telco tag text file:** For telco services only, you can create service-level provisioning tags by defining them in a text file and then loading the file into BRM. You can include supplementary services and ERAs in the tags.

You can also create account-level ERAs in this same file.

See "Defining Provisioning Tags for Telco Services Using the `pin_telco_tags` File" in *BRM Telco Integration*.

- **Provisioning policy source file:** You can define product provisioning tags by editing and compiling the policy file that is the source file for all provisioning operations.

Important: The provisioning tag framework is the preferable method for creating provisioning tags.

See "[Using a Policy Source File to Set Up Provisioning](#)".

Deciding Which Provisioning Tag Method to Use

For telco services, you should create provisioning tags by using one of these methods. The order of preference is shown below.

1. Provisioning tag framework. See "[Using the Provisioning Tag Framework](#)".
2. Provisioning Tags application in Pricing Center.
3. Telco tag text file.
4. Provisioning policy source file. See "[Using a Policy Source File to Set Up Provisioning](#)".

For non-telco services, you should create provisioning tags by using one of these methods. The order of preference is shown below:

1. Provisioning tag framework. See "[Using the Provisioning Tag Framework](#)".
2. Provisioning policy source file. See "[Using a Policy Source File to Set Up Provisioning](#)".

Using the Provisioning Tag Framework

You can create service-level or account-level provisioning tags using the provisioning tag framework. This framework stores provisioning tags in the `/config/provisioning_tag` object.

Important: To create most provisioning tags for telco services, use the Provisioning Tags application in Pricing Center or the `load_pin_telco_tags` utility.

You define a provisioning tag by specifying the services to which it applies and the opcodes to run when the product or discount containing the tag is purchased or canceled. You might need to create custom opcodes for some provisioning tags.

To create a provisioning tag using this framework, do the following:

- Configure the provisioning tag in the `pin_config_provisioning_tags.xml` file. See ["Configuring Provisioning Tags"](#).
- Load the provisioning tag configuration into the `/config/provisioning_tag` object with the `load_config_provisioning_tags` utility. ["Loading Provisioning Tag Configurations"](#).
- Add the tag to the provisioning policy source file and compile the file. You do this if you will include the provisioning tag in a product and if the tag uses a service associated with the `__DEFAULT__` provisioning tag. See ["Modifying and Compiling the Provisioning Policy Source File"](#).

Configuring Provisioning Tags

To create provisioning tags, you configure the provisioning tags configuration file, `pin_config_provisioning_tags.xml`. This file is located in the `BRM_Home/sys/data/config` directory.

To define a provisioning tag in this configuration file, you specify the following:

- A unique name for the provisioning tag.

Note: The provisioning tag for a product or discount must be the name of the offer profile with which the product or discount is associated.

- For a service-level tag, the permitted services.
- For an account-level tag, `/account`.
- The name and number of each opcode to run when the product or discount containing the provisioning tag is purchased or canceled. These opcodes contain the business logic to perform the actual provisioning, such as creating a profile.
- Parameters that specify the fields to be added to each opcode's input list and the value for each field.

Important:

- Do not remove existing provisioning tags from `pin_config_provisioning_tags.xml` when adding new tags unless you want existing tags removed from the `/config/provisioning_tag` objects in the database.
 - When you load `pin_config_provisioning_tags.xml`, all existing instances of `/config/provisioning_tag` are removed from the database, so only the provisioning tags defined in the file when you load it will be in the database.
-
-

[Table 5-1](#) lists the elements in the `pin_config_provisioning_tags.xml` file, the syntax to use for each element, and a description of how to specify each element. The syntax is based on the default version of the file:

Table 5–1 Elements in pin_config_provisioning_tags XML File

Element	Syntax	Description
BusinessConfiguration	<pre><BusinessConfiguration xmlns="http://www.portal.com/ schemas/BusinessConfig" xmlns:xsi="http://www.w3.org/ 2001/XMLSchema-instance" xsi:schemaLocation="http://www.po rtal.com/schemas/ BusinessConfig business_ configuration.xsd"></pre>	The root element common to all BRM configurations.
ProvisioningTag Configuration	<pre><ProvisioningTagConfiguration></pre>	Opens the provisioning tag configuration. Contains the ProvisioningTagList element.
ProvisioningTag List	<pre><ProvisioningTagList></pre>	Contains all the provisioning tag definitions.
ProvisioningTag	<pre><ProvisioningTag name= "__DEFAULT__"></pre>	Contains the definition of a provisioning tag and specifies the tag's name.
PermittedTypes	<pre><PermittedTypes>/service/ip </PermittedTypes></pre>	Specifies a service or account that is valid for the provisioning tag. One or more of these tags can be part of a provisioning tag definition. When the API for getting a list of provisioning tags is called, the array of permitted types is looked up and only provisioning tags applicable to the specific service type are returned.
OpcodeList	<pre><OpcodeList></pre>	Contains the definition of one opcode. The opcodes specified for a provisioning tag contain the business logic required for provisioning when purchasing and canceling a product. A provisioning tag can include multiple opcodes.
OpcodeName	<pre><OpcodeName> PCM_OP_SUBSCRIPTION_POL_PURCHASE_ PROD_PROVISIONING </OpcodeName></pre>	Specifies the opcode's name. See "Specifying an Opcode in a Provisioning Tag to Create an ERA" .
OpcodeNumber	<pre><OpcodeNumber>417 </OpcodeNumber></pre>	Specifies the hard-coded number for an opcode. To find an opcode's number, see the opcode header files in the <i>BRM_Home/include/ops</i> directory.

Table 5–1 (Cont.) Elements in pin_config_provisioning_tags XML File

Element	Syntax	Description
OpcodeMode	<OpcodeMode>0</OpcodeMode>	Indicates when the opcode should be called, as follows: <ul style="list-style-type: none"> ■ 0: on product or discount purchase ■ 1: on product or discount cancellation ■ 2: on both purchase and cancellation
OpcodeParamsList	<OpcodeParamsList>	Defines a parameter for an opcode. You can have multiple parameters. Each parameter is a name-value pair.
OpcodeParamName	<OpcodeParamName>PIN_FLD_POID</OpcodeParamName>	Specifies a field name to be added to the input flist. If the field is part of a substruct or array, use a period to separate the substruct or array name and the field name. For example: PIN_FLD_EVENT.PIN_FLD_ACCOUNT_OBJ
OpcodeParamValue	<OpcodeParamValue>\$\$SERVICE\$</OpcodeParamValue>	Specifies the value of the field. For certain values that are not known until run time, you can use a variable, such as \$\$SERVICE\$. See " Variables for Parameter Values ".

For an example of the XML syntax, see "[Sample Provisioning Tag XML File](#)".

Specifying an Opcode in a Provisioning Tag to Create an ERA

When defining a configuration for the provisioning tag framework, you specify one or more opcodes to perform an action, such as creating an ERA. You can specify that an opcode run when a product or discount is purchased, canceled, or both. You can use an existing opcode or design a custom opcode.

If the provisioning tag is designed to create an ERA, you can specify that the PCM_OP_SUBSCRIPTION_PROVISION_ERA opcode run at both purchase and cancellation time. This opcode creates, modifies, or deletes a profile (**/profile** object). ERAs are stored in profiles.

The actions PCM_OP_SUBSCRIPTION_PROVISION_ERA takes depends on the value specified for the PIN_FLD_ACTION parameter in the provisioning tag:

- If PIN_FLD_ACTION is **Purchase**, the opcode checks if a profile already exists. If the profile does not exist, the opcode calls PCM_OP_CUST_CREATE_PROFILE to create the profile. If the profile does exist, the opcode calls PCM_OP_CUST_MODIFY_PROFILE to add data passed in from the input flist and to increment the reference counter by 1.
- If PIN_FLD_ACTION is **Cancel**, the opcode decrements the reference counter by 1. If the counter is 0, the opcode calls PCM_OP_CUST_DELETE_PROFILE to delete the profile.

For example, you can create a friends and family ERA for a service by calling the `PCM_OP_SUBSCRIPTION_PROVISION_ERA` at purchase time with the parameters in [Table 5-2](#):

Table 5-2 *PCM_OP_SUBSCRIPTION_PROVISION_ERA Parameters*

OpcodeParamName	OpcodeParamValue
PIN_FLD_POID	0.0.0.0 /profile/serv_extrating -1
PIN_FLD_ACCOUNT_OBJ	\$ACCOUNT\$
PIN_FLD_FLAGS	0
PIN_FLD_SERVICE_OBJ	\$SERVICE\$
PIN_FLD_STR_VAL	12, 13
PIN_FLD_NAME	FRIENDS_FAMILY
PIN_FLD_INHERITED_INFO.	MYFRIENDS
PIN_FLD_EXTRATING.	
PIN_FLD_LABEL	

These name-value pairs indicate that an ERA named `FRIENDS_FAMILY` and an ERA label named `MYFRIENDS` are created. Because both the account and service POIDs are specified, the opcode creates a service-level profile (**/profile/serv_extrating**) object. If the service POID was not specified, the opcode would create an account-level profile (**/profile/acct_extrating**) object.

Note:

- `PIN_FLD_POID` is the POID, in string format, of the object the provisioning tag will create. This is converted to a POID when the opcode runs. If you are using multiple database schemas, the string is converted to the correct schema database number.
 - For a service-level profile, the POID type is **/profile/serv_extrating**, as in the previous example. For an account-level profile, the POID type is **/profile/acct_extrating**.
 - `PIN_FLD_FLAGS` specifies that `PCM_OP_SUBSCRIPTION_PROVISION_ERA` is called at purchase time.
-
-

You must include the opcode twice in a provisioning tag, once with `PIN_FLD_FLAGS` set to 0 and once with `PIN_FLD_FLAGS` set to 1, so that it runs both at purchase and cancellation time.

- `PIN_FLD_STR_VAL` specifies that the profile name and profile description are localized and are stored in the **/string** object under string IDs 12 and 13.
- `PIN_FLD_ACCOUNT_OBJ` and `PIN_FLD_SERVICE_OBJ` use variables. See ["Variables for Parameter Values"](#).

Variables for Parameter Values

You can use the following variables in [Table 5-3](#) to specify certain values available only at the time the opcode is run:

Table 5–3 Run-Time Variables for Parameters

Variable	Description
\$ACCOUNT\$	Account POID
\$SERVICES\$	Service POID
\$PRODUCT\$	Product POID
\$DISCOUNT\$	Discount POID
\$OFFERING\$	POID of the purchased product or discount
\$PROVTAG\$	Provisioning tag

Default Provisioning Tag

The default `/config/provisioning_tag` object contains the `__DEFAULT__` provisioning tag. This tag is defined in the default `pin_config_provisioning_tags.xml` file. The tag calls the following opcodes:

- `PCM_OP_SUBSCRIPTION_POL_PURCHASE_PROD_PROVISIONING`, on product purchase.
- `PCM_OP_SUBSCRIPTION_POL_CANCEL_PROD_PROVISIONING`, on product cancellation.

If you have customized these opcodes, they set and clear fields in `/service` objects when a product is purchased and canceled. If you have not customized these opcodes, you do not need to use them. You can specify other opcodes in the provisioning tags file to perform necessary actions.

The `__DEFAULT__` provisioning tag is always called for specified services when a product is purchased or canceled, so you do not need to include these opcodes in any other provisioning tags you define. See the default `pin_config_provisioning_tags.xml` file for the list of services.

Using Custom Opcodes

You can create custom opcodes to perform actions not supported by existing opcodes and specify the custom opcodes in the provisioning tags configuration file. For example, you can write an opcode to add fields to a `/service` object.

Configuring Provisioning Tags for Policy-Driven Charging

Configure the resources that BRM should track by creating provisioning tags for your services. Each provisioning tag contains the types of resources valid for that tag and information on the policy attributes which are used in provisioning policy for that resource.

Policy attributes are the configured characteristics associated with a provisioning policy. They contain the information you must provide to the specific opcode which is to process changes to the product or discount associated with that provisioning tag (for example, an offer profile threshold breach notification is required and/or the language in which the subscriber requires the notification).

Note: BRM stores policy attributes as subscriber preferences (`/profile/subscriber_preferences` objects containing the following elements:

- Name (required)
 - Type (optional)
 - Value (optional)
-
-

See "Managing Subscriber Preferences" in *BRM Telco Integration* for information on storing subscriber preferences.

About the Default Provisioning Tag Provided by BRM

BRM provides a default provisioning tag in the `pin_offer_profile_provisioning_tags_policy_attributes.xml` file located in `BRM_Home/sys/data/config` directory.

The first section of the definition seen in [Example 5-1](#) shows the valid resources for the default provisioning tag. It is followed by the information on the opcodes to be called when the product or discount containing the provisioning tag is purchased or canceled and the parameters that specify the fields to be added to the opcode's input list and the value for each field.

Collecting Data for Provisioning Tags

Collect the following data for any additional services that BRM should track for policy-driven charging.

- The name for your provisioning tag. This must be the name of the offer profile with which you associate this provisioning configuration.
- Permitted types of services
- Subscriber preferences for the service type stored in the database as `/profile/subscriber_preferences` objects.

For more information see "Managing Subscriber Preferences" in *BRM Telco Integration*.

- Opcodes that are to be used in association with the provisioning tag
You must set up the following for each of the opcodes you include in a provisioning tag:
 - The name and number of each opcode to run and when the opcode must be run. These opcodes contain the business logic to perform the actual provisioning, such as creating a profile.
 - Parameters that specify the fields to be added to each opcode's input list and the value for each field.

This information can now be configured in the `pin_offer_profile_provisioning_tags_policy_attributes.xml` file for use with your offer profiles.

Configuring Provisioning Tags for Offer Profiles

To configure provisioning tags for use with your offer profiles:

1. Go to the `BRM_Home/sys/data/config` directory.

2. Open the `pin_offer_profile_provisioning_tags_policy_attributes.xml` file in an XML editor or text editor.

Tip: Save a copy of the default configuration file before you make any changes to it.

3. Do one of the following to update this file:
 - Edit the default provisioning tag contained in this file.
 - Add your provisioning tag by placing your definition just below the default provisioning tag (*Platinum*). Include the following for each provisioning tag:

- Provisioning tag name

```
<ProvisioningTag name="YourProvisioningTagName">
```

where *YourProvisioningTagName* is the name of your offer profile.

- Each service that is valid for the provisioning tag as a **PermittedType** element as shown in [Example 5-1](#).
- Specifications for the opcodes to call for this provisioning tag as shown in [Example 5-1](#).

Tip: [Table 5-1](#) lists the elements in the `pin_offer_profile_provisioning_tags_policy_attributes.xml` file, the syntax to use for each element, and a description of how to specify each element.

4. Save the `pin_offer_profile_provisioning_tags_policy_attributes.xml` file.

Modifying Provisioning Tags

You use provision tags as the names of offer profiles associated with products and discounts. If you modify a provision tag that is in use as an offer profile, be sure to modify the corresponding offer profile name accordingly.

See "[Policy-Driven Charging](#)" for information on offer profiles.

Important:

- Do not remove existing provisioning tags from `pin_config_provisioning_tags.xml` when adding new tags unless you want existing tags removed from the `/config/provisioning_tag` objects in the database.
 - When you load `pin_config_provisioning_tags.xml`, all existing instances of `/config/provisioning_tag` are removed from the database, so only the provisioning tags defined in the file when you load it will be in the database.
-
-

Loading Provisioning Tag Configurations

If you are loading provisioning tags for policy-driven charging, see "[Loading Provisioning Tags for Policy-Driven Charging](#)".

After you configure provisioning tags in the `pin_config_provisioning_tags.xml` file, load the tags into the database with the `load_config_provisioning_tags` utility.

Caution: The utility that loads provisioning tags into the database overwrites existing provisioning tags. When updating provisioning tags, you cannot load new provisioning tags only. You must load the complete set of provisioning tags each time you run the utility.

To load provisioning tag configurations:

1. Go to the directory in which the `pin_config_provisioning_tags.xml` file is located. The default location is `BRM_Home/sys/data/config`.
2. Use the following command to run the `load_config_provisioning_tags` utility:

```
load_config_provisioning_tags pin_config_provisioning_tags.xml
```

Important:

- When you run the utility, the `pin_config_provisioning_tags.xml` and `business_configuration.xsd` files must be in the same directory. By default, both files are in `BRM_Home/sys/data/config`.
 - This utility needs a configuration (`pin.conf`) file in the directory from which you run the utility.
-
-

If you do not run the utility from the directory in which `pin_config_provisioning_tags.xml` is located, include the complete path to the file, for example:

```
load_config_provisioning_tags BRM_Home/sys/data/config/pin_config_provisioning_tags.xml
```

For more information, see `load_config_provisioning_tags`.

3. Stop and restart the Connection Manager (CM).
4. To verify that the provisioning tags were loaded, display the `/config/provisioning_tag` object by using the Object Browser or the `robj` command with the `testnap` utility.

Loading Provisioning Tags for Policy-Driven Charging

Use the `load_config_provisioning_tags` utility to load the `pin_offer_profile_provisioning_tags_policy_attributes.xml` file into the BRM database.

To do so:

1. Ensure that the provisioning tags required for policy-driven charging are configured in the `pin_offer_profile_provisioning_tags_policy_attributes.xml` XML file.
2. Go to the directory in which the `pin_offer_profile_provisioning_tags_policy_attributes.xml` file is located. The default location is `BRM_Home/sys/data/config`.
3. Commit the provisioning tags and policy attributes XML file to the BRM database.

```
load_config_provisioning_tags -d -v pin_offer_profile_provisioning_tags_policy_attributes.xml
```

where:

- `-d` creates a log file for debugging purposes.

- `-v` displays information about successful or failed processing as the utility runs

For more information on the `load_config_provisioning_tags` utility, see "load_config_provisioning_tags" in *BRM Developer's Guide*.

4. To verify that the provisioning tags were loaded, display the `/config/provisioning_tag` object by using the Object Browser or the `robj` command with the `testnap` utility.
5. Stop and restart the Connection Manager (CM).

The provisioning tags and policy attributes information associated with policy-driven charging is now stored in `/config/provisioning_tag` objects in the database.

Modifying and Compiling the Provisioning Policy Source File

If a provisioning tag defined in the provisioning tag framework uses the `PCM_OP_SUBSCRIPTION_POL_PURCHASE_PROD_PROVISIONING` policy opcode, you must add the tag to the `fm_subscription_pol_provisioning.c` file and recompile the file.

A provisioning tag uses `PCM_OP_SUBSCRIPTION_POL_PURCHASE_PROD_PROVISIONING` opcode if it is included in a product and if it uses a service associated with the `__DEFAULT__` provisioning tag. See "Sample Provisioning Tag XML File" for the list of permitted services for `__DEFAULT__`.

Modifying and compiling the provisioning policy source file enables `PCM_OP_SUBSCRIPTION_POL_PURCHASE_PROD_PROVISIONING` to handle the tags.

To modify the provisioning policy source file for provisioning tags created using the provisioning tag framework:

1. Open the `BRM_Home/source/sys/fm_subscription_pol/fm_subscription_pol_provisioning.c` file.
2. Add the provisioning tag name to the `service_info` table for the appropriate service.

For example, to add a provisioning tag called `test` to `/service/ip`, change this code:

```
static char *tags_ip[] = {
    "example",
    NULL /* MUST BE LAST! */
};
```

to the following:

```
static char *tags_ip[] = {
    "example",
    "test",
    NULL /* MUST BE LAST! */
};
```

3. Add code in the `plp` function to handle the new tag. You do this in the `PROVISIONING FUNCTIONS` section. The functions are grouped by service type.

For example, to add the provisioning tag `test` to `/service/ip`, change this code:

```
static void
plp_ip(pcm_context_t *ctxp, poid_t *svc_obj_p, int32 buy,
char *tag, pin_errbuf_t *ebufp)
{
    if (strcmp(tag, "example") == 0) {
        plp_example(ctxp, svc_obj_p, buy, tag, ebufp);
    }
}
```

```
    }
    else{
        plp_ssg(ctxp, svc_obj_p, buy, tag, ebufp);
    }
}
```

to the following:

```
static void
plp_ip(pcm_context_t *ctxp, poid_t *svc_obj_p, int32 buy,
char *tag, pin_errbuf_t *ebufp)
{
    if (strcmp(tag, "example") == 0) {
        plp_example(ctxp, svc_obj_p, buy, tag, ebufp);
    }
    else if (strcmp(tag, "test") == 0) {
        /*skip*/
    }else{
        plp_ssg(ctxp, svc_obj_p, buy, tag, ebufp);
    }
}
```

4. Compile and save the file.

Using a Policy Source File to Set Up Provisioning

You can define provisioning tags directly in the `fm_subscription_pol_provisioning.c` file without using the provisioning tag framework. This file is the single source file for all provisioning operations.

Important: The provisioning tags framework is the preferable method for creating provisioning tags.

To define provisioning tags directly in source code, follow these steps:

1. Open the `BRM_Home/source/sys/fm_subscription_pol/fm_subscription_pol_provisioning.c` file.
2. Define your provisioning tags by following the instructions in the file.
 - a. In each entry in the `provisioning_tags` table, include the name of the service associated with the tag, the tag name, and calls to service-specific functions.
 - b. Ensure that each function included in the table does the following:
 - Changes fields in the service object when customers purchase the service
 - Clears the appropriate fields when customers cancel the service
3. Compile and save the file.

If you create a provisioning tag in the policy source file, you must modify and recompile the source file to modify or delete the tag.

Sample Provisioning Tag XML File

Following is the default provisioning tag XML file. This file defines the provisioning tag named `__DEFAULT__`, which includes several permitted services and two opcodes:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```

<BusinessConfiguration
  xmlns="http://www.portal.com/schemas/BusinessConfig"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.portal.com/schemas/BusinessConfig business_
configuration.xsd">

  <ProvisioningTagConfiguration>
  <ProvisioningTagList>

    <ProvisioningTag name="__DEFAULT__">

      <PermittedTypes>/service/email</PermittedTypes>
      <PermittedTypes>/service/ip/gprs</PermittedTypes>
      <PermittedTypes>/service/content</PermittedTypes>
      <PermittedTypes>/service/vpdn</PermittedTypes>
      <PermittedTypes>/service/ip</PermittedTypes>
      <PermittedTypes>/service/ip/gprs</PermittedTypes>
      <PermittedTypes>/service/telco</PermittedTypes>
      <PermittedTypes>/service/telco/gsm</PermittedTypes>
      <PermittedTypes>/service/telco/gsm/data</PermittedTypes>
      <PermittedTypes>/service/telco/gsm/fax</PermittedTypes>
      <PermittedTypes>/service/telco/gsm/sms</PermittedTypes>
      <PermittedTypes>/service/telco/gsm/telephony</PermittedTypes>

      <OpcodeList>
        <OpcodeName>PCM_OP_SUBSCRIPTION_POL_PURCHASE_PROD_
PROVISIONING</OpcodeName>
        <OpcodeNumber>417</OpcodeNumber>
        <OpcodeMode>0</OpcodeMode>

        <OpcodeParamsList>
          <OpcodeParamName>PIN_FLD_POID</OpcodeParamName>
          <OpcodeParamValue>$SERVICE$</OpcodeParamValue>
        </OpcodeParamsList>
        <OpcodeParamsList>
          <OpcodeParamName>PIN_FLD_PROVISIONING_TAG</OpcodeParamName>
          <OpcodeParamValue>$PROVTAG$</OpcodeParamValue>
        </OpcodeParamsList>
      </OpcodeList>

      <OpcodeList>
        <OpcodeName>PCM_OP_SUBSCRIPTION_POL_CANCEL_PROD_
PROVISIONING</OpcodeName>
        <OpcodeNumber>418</OpcodeNumber>
        <OpcodeMode>1</OpcodeMode>

        <OpcodeParamsList>
          <OpcodeParamName>PIN_FLD_POID</OpcodeParamName>
          <OpcodeParamValue>$SERVICE$</OpcodeParamValue>
        </OpcodeParamsList>
        <OpcodeParamsList>
          <OpcodeParamName>PIN_FLD_PROVISIONING_TAG</OpcodeParamName>
          <OpcodeParamValue>$PROVTAG$</OpcodeParamValue>
        </OpcodeParamsList>
      </OpcodeList>

    </ProvisioningTag>
  </ProvisioningTagList>

```

```

    </ProvisioningTagConfiguration>

</BusinessConfiguration>

```

Default Provisioning Tag for Policy-Driven Charging

Example 5–1 shows the default provisioning tag provided by BRM in the `pin_offer_profile_provisioning_tags_policy_attributes.xml` file located in the `BRM_Home/sys/data/config` directory.

Example 5–1 Default Provisioning Tag Configuration

```

<BusinessConfiguration xsi:schemaLocation="http://www.portal.com/schemas/BusinessConfig business_
configuration.xsd">
<ProvisioningTagConfiguration>
  <ProvisioningTagList>
    <ProvisioningTag name="Platinum">
      <PermittedTypes>/service/email</PermittedTypes>
      <PermittedTypes>/service/ip/gprs</PermittedTypes>
      <PermittedTypes>/service/content</PermittedTypes>
      <PermittedTypes>/service/vpdn</PermittedTypes>
      <PermittedTypes>/service/ip</PermittedTypes>
      <PermittedTypes>/service/fax</PermittedTypes>
      <PermittedTypes>/service/ip/gprs</PermittedTypes>
      <PermittedTypes>/service/telco</PermittedTypes>
      <PermittedTypes>/service/telco/gsm</PermittedTypes>
      <PermittedTypes>/service/telco/gsm/data</PermittedTypes>
      <PermittedTypes>/service/telco/gsm/fax</PermittedTypes>
      <PermittedTypes>/service/telco/gsm/sms</PermittedTypes>
      <PermittedTypes>/service/telco/gsm/sms</PermittedTypes>
      <OpcodeList>
        <OpcodeName>PCM_OP_CUST_SET_SUBSCRIBER_PREFERENCES</OpcodeName>
        <OpcodeNumber>3916</OpcodeNumber>
        <OpcodeMode>0</OpcodeMode>
        <OpcodeParamsList>
          <OpcodeParamName>PIN_FLD_POID</OpcodeParamName>
          <OpcodeParamValue>0.0.0.0 /profile/subscriber_preferences -1</OpcodeParamValue>
        </OpcodeParamsList>
        <OpcodeParamsList>
          <OpcodeParamName>PIN_FLD_DELETED_FLAG</OpcodeParamName>
          <OpcodeParamValue>0</OpcodeParamValue>
        </OpcodeParamsList>
        <OpcodeParamsList>
          <OpcodeParamName>PIN_FLD_ACCOUNT_OBJ</OpcodeParamName>
          <OpcodeParamValue>$ACCOUNT$</OpcodeParamValue>
        </OpcodeParamsList>
        <OpcodeParamsList>
          <OpcodeParamName>PIN_FLD_SERVICE_OBJ</OpcodeParamName>
          <OpcodeParamValue>$SERVICE$</OpcodeParamValue>
        </OpcodeParamsList>
        <OpcodeParamsList>
          <OpcodeParamName>PIN_FLD_NAME</OpcodeParamName>
          <OpcodeParamValue>SET_SUBSCRIBER_PREFERENCES</OpcodeParamValue>
        </OpcodeParamsList>
        <OpcodeParamsList>
          <OpcodeParamName>PIN_FLD_SUBSCRIBER_PREFERENCES[0].PIN_FLD_NAME</OpcodeParamName>
          <OpcodeParamValue>Language</OpcodeParamValue>
        </OpcodeParamsList>
        <OpcodeParamsList>
          <OpcodeParamName>PIN_FLD_SUBSCRIBER_PREFERENCES[0].PIN_FLD_

```

```

VALUE</OpcodeParamName>
    <OpcodeParamValue>English</OpcodeParamValue>
  </OpcodeParamsList>
</OpcodeParamsList>
  <OpcodeParamName>PIN_FLD_SUBSCRIBER_PREFERENCES[0].PIN_FLD_SUBSCRIBER_PREFERENCE_
ID</OpcodeParamName>
  <OpcodeParamValue>1</OpcodeParamValue>
  </OpcodeParamsList>
</OpcodeParamsList>
  <OpcodeParamName>PIN_FLD_SUBSCRIBER_PREFERENCES[1].PIN_FLD_
NAME</OpcodeParamName>
  <OpcodeParamValue>Channel</OpcodeParamValue>
  </OpcodeParamsList>

  <OpcodeParamsList>
    <OpcodeParamName>PIN_FLD_SUBSCRIBER_PREFERENCES[1].PIN_FLD_
VALUE</OpcodeParamName>
    <OpcodeParamValue>IVR</OpcodeParamValue>
  </OpcodeParamsList>
</OpcodeParamsList>
  <OpcodeParamName>PIN_FLD_SUBSCRIBER_PREFERENCES[1].PIN_FLD_SUBSCRIBER_PREFERENCE_
ID</OpcodeParamName>
  <OpcodeParamValue>2</OpcodeParamValue>
  </OpcodeParamsList>
</OpcodeList>
<OpcodeList>
  <OpcodeName>PCM_OP_CUST_SET_SUBSCRIBER_PREFERENCES</OpcodeName>
  <OpcodeNumber>3916</OpcodeNumber>
  <OpcodeMode>1</OpcodeMode>
  <OpcodeParamsList>
    <OpcodeParamName>PIN_FLD_POID</OpcodeParamName>
    <OpcodeParamValue>0.0.0.0 /profile/subscriber_preferences -1</OpcodeParamValue>
  </OpcodeParamsList>
  <OpcodeParamsList>
    <OpcodeParamName>PIN_FLD_DELETED_FLAG</OpcodeParamName>
    <OpcodeParamValue>1</OpcodeParamValue>
  </OpcodeParamsList>
  <OpcodeParamsList>
    <OpcodeParamName>PIN_FLD_ACCOUNT_OBJ</OpcodeParamName>
    <OpcodeParamValue>$ACCOUNT$</OpcodeParamValue>
  </OpcodeParamsList>
  <OpcodeParamsList>
    <OpcodeParamName>PIN_FLD_SERVICE_OBJ</OpcodeParamName>
    <OpcodeParamValue>$SERVICE$</OpcodeParamValue>
  </OpcodeParamsList>
  <OpcodeParamsList>
    <OpcodeParamName>PIN_FLD_NAME</OpcodeParamName>
    <OpcodeParamValue>DEL_SUBSCRIBER_PREFERENCES</OpcodeParamValue>
  </OpcodeParamsList>
</OpcodeList>
</ProvisioningTag>
</ProvisioningTagList>
</ProvisioningTagConfiguration>
</BusinessConfiguration>

```

Working with Extended Rating Attributes

This chapter describes how to use extended rating attributes (ERAs) in your Oracle Communications Billing and Revenue Management (BRM) system and how BRM rates events using ERAs.

About Extended Rating Attributes

Extended rating attributes (ERAs) provide special rates or discounts based on a specific attribute of a service or account, such as a telephone number. For example, you use ERAs to offer special friends and family rates or a birthday discount.

You can create two types of ERAs:

- **Service-level:** An ERA that is used with a specific service, such as a telco service. Special rates for friends and family and closed user group telephone calls are examples of service-level ERAs.

Note: A service-level ERA can be assigned to a service that represents a subscription so that it applies to all services associated with a customer's subscription.

- **Account-level:** An ERA that can be used with any service (for example, a birthday discount).

You create an ERA by adding it to a provisioning tag. See "[About Provisioning Tags](#)".

ERAs must be configured with specific data for each customer. For example, to set up a birthday discount, you must specify the customer's birthday.

You configure ERAs in Customer Center, where ERAs are called *promotions*. ERAs you configure are stored as profiles.

If you are using ERAs with telco services, see "About extended rating attributes for telco services" in *BRM Telco Integration*.

For more information about ERAs, see:

- [About Multiple ERA Lists](#)
- [About Configuring ERAs in Customer Center](#)
- [About Sharing ERAs](#)
- [Creating ERAs](#)

Note:

- ERAs are considered when BRM checks for rating and discount criteria as part of real-time rating. By default, BRM includes all ERAs in this process, but you can improve real-time rating performance by configuring BRM to omit account-level ERAs, service-level ERAs, or both from the discount criteria. For details, see "Filtering the ERAs considered during rating and discounting" in *BRM System Administrator's Guide*.
- For service-level ERAs, be aware that ERA profile information is not automatically transferred between plans during plan transition or generation change. If the two plans have some common provisioning tags, the ERA profile information can be reconfigured in the new plan.

About Multiple ERA Lists

For some types of ERAs, such as friends and family, you can create multiple lists for the same ERA. Each list is identified as a *label* in the BRM system. This enables you to set up rating and discounting to select the rate or discount based on the label name.

For example, you can design a friends and family ERA for a GSM telephony service that charges different rates for friends and family using the labels in [Table 6-1](#):

Table 6-1 Example Labels and Rates

ERA Label	Rate
MYFAMILY	1 cent per minute
MYFRIENDS	2 cents per minute

You include labels when you define the ERA in a provisioning tag.

For more information about the different methods of defining provisioning tags, see "[About Provisioning Tags](#)". For an example of defining ERA labels using the provisioning tag framework, see "[Sample Friends and Family Provisioning Tag](#)".

You use price model selectors, discount model selectors, or rate plan selectors to specify different rates or discounts for each label. See "[Using ERAs with Multiple Lists](#)".

About Configuring ERAs in Customer Center

Use Customer Center to configure an ERA for an individual customer. When a customer purchases a product or discount that includes a provisioning tag with an ERA, that ERA is displayed in Customer Center as a *promotion*. You specify values for the ERA on the **Promotion** tab.

When you configure an ERA in Customer Center, the values are stored in one of these profile objects:

- `/profile/serv_extrating` for a service-level ERA
- `/profile/acct_extrating` for an account-level ERA

Note:

- Even though an account is qualified to use ERAs, you do not have to implement them in the account.
 - BRM does not validate any data entered when configuring ERAs (for example, telephone numbers for the friends and family discount). To create validation rules for these entries, edit the PCM_OP_CUST_POL_PREP_PROFILE policy opcode.
 - ERA codes are defined in Pipeline Manager configuration files. The types of values you enter in Customer Center depends on how you configure the ERAs in Pipeline Manager. For example, if the ERA is configured to match a telephone number in an event, you would enter telephone numbers in Customer Center.
 - When configuring ERAs in Customer Center, use only uppercase letters, ASCII 7-bit punctuation, and no spaces.
-
-

About Sharing ERAs

You can share the ERA values you configure for one service with other services by using profile sharing groups. For example, you can share the same list of phone numbers for a friends and family ERA among all the customers whose phone numbers are included in the list.

You can share service-level ERAs only.

For more information, see "Working with profile sharing groups" in *BRM Managing Customers*.

Creating ERAs

This section is an overview of the main steps you take to create an ERA. To create a friends and family ERA, see "[Creating Friends and Family ERAs](#)".

Each of the following steps is a separate task. Use the links for detailed information about each task:

1. **Create a provisioning tag.** ERAs are defined in provisioning tags, which you can create in a few different ways. See "[About Provisioning Tags](#)".
2. **Define how the ERA is rated.** Configure pipeline rating or real-time rating as needed to rate events based on the ERA. For example, an ERA can be rated based on the usage type or on the ERA label name.

See "[About Rating Based on Friends and Family ERAs](#)", "[Configuring the IRL_UsageType iRule for ERAs](#)", and "[Rating an Event Based on Extended Rating Attributes](#)".

3. **Add the ERA's provisioning tag to a product or discount.**

Important: You cannot use provisioning tags created with the Services Framework Manager with discounts. See "[About provisioning tags for telco services](#)" in *BRM Telco Integration*.

4. **Add the name and description to the ERA description file.** The name and description appears in Customer Center. For provisioning tags created with the

provisioning tag framework, you must take additional steps to set up the name and description. See "[Customizing ERA Names and Descriptions for Client Applications](#)".

5. **Configure the ERA for an individual customer.** See "[About Configuring ERAs in Customer Center](#)".
6. **(Optional) Create a profile sharing group to share the ERA configuration with other accounts or services.** See "[About Sharing ERAs](#)".

Important: You can share service-level ERAs only.

Creating Friends and Family ERAs

BRM has built-in support for friends and family ERAs. This feature enables you to create an ERA with multiple lists, each containing phone numbers, APN addresses, email addresses, or other values. You can also use profile sharing to share a friends and family ERA with multiple services.

Following is a description of the steps for creating a friends and family ERA. Use the links for more detailed information about carrying out each task:

1. Create a provisioning tag.

Create a provisioning tag with a service-level ERA called FRIENDS_FAMILY and one or more labels, which are ERA lists. The labels are optional, but they give you the flexibility to define multiple lists for an ERA and to use the label name as a rule for model selectors or rate plan selectors.

You create the provisioning tag in different ways depending on its service:

- For any service, you can use the provisioning tag framework. This involves defining the tag in the **pin_config_provisioning_tags.xml** file.

The definition of the tag includes one or more opcodes that are run when the provisioning tag is purchased. For ERAs, you can use PCM_OP_SUBSCRIPTION_PROVISION_ERA. This opcode creates a **/profile** object for the ERA.

See "[Using the Provisioning Tag Framework](#)".

- For telco services, use the Provisioning Tags application in Pricing Center, or modify and load the service-specific version of the **pin_telco_tags** file.

Important: For telco services, you cannot use the provisioning tag framework to include supplementary services and service extensions in the same provisioning tag with the ERA.

2. Enable pin_notify entries.

- a. Uncomment the following entries in the **pin_notify** file:

```
3788 0 /event/group/sharing/profiles/create
3788 0 /event/group/sharing/profiles/modify
```

- b. Load the **pin_notify** file and restart the Connection Manager (CM).

3. For pipeline rating, configure the ISC_ProfileAnalyzer iScript if needed.

ISC_ProfileAnalyzer matches a value in an EDR, such as a phone number, with lists in the friends and family ERA. It then adds the names of ERA labels that include that value to the EDR.

By default, this iScript analyzes an ERA named FRIENDS_FAMILY that uses a GSM telephony service. You can customize the iScript to handle a different service or ERA name.

To enable ISC_ProfileAnalyzer, configure the FCT_iScript module and set the **Active** entry to **True**.

Important: For rerating, ISC_ProfileAnalyzer must be enabled. The default registry file is *BRM_Home/conf/wirelessRealtime.reg*.

See "[Customizing Pipeline Rating for Friends and Family ERAs](#)".

4. For real-time rating, customize the PCM_OP_RATE_POL_PROCESS_ERAS policy opcode if needed.

PCM_OP_RATE_POL_PROCESS_ERAS compares a value in an event, such as a phone number, with lists in the friends and family ERA. It then adds the names of the ERA and ERA label that include that value to the event.

By default, the opcode is configured to work with a GSM service and to compare telephone numbers called from and called to. You can customize the opcode to work with a different service and to compare different values from an event with the ERA values.

See "[Customizing Real-Time Rating for Friends and Family ERAs](#)".

5. For pipeline rating, create a price model selector or discount model selector.
For pipeline events, use a price or discount model selector to specify different rates or discounts for different ERA labels. See "[Using ERAs with Multiple Lists](#)".
6. For real-time rating, create a rate plan selector.
For real-time events, use a rate plan selector to specify different rates for different ERA labels. Use the event attribute EVENT.PIN_FLD_PROFILE_LABEL_LIST in the rate plan selector matrix to specify different values for different label names.
7. Create a product or discount that includes the provisioning tag.
For a product, include the rate plan with the price model selector or rate plan selector you defined. For a discount, include the discount model selector you defined.

Important:

- You cannot use provisioning tags created with the Services Framework Manager with discounts.
 - Use the provisioning tag framework to create an ERA that you want to add directly to a discount. See "[Using the Provisioning Tag Framework](#)".
-

8. Add the names and descriptions to the ERA description file.

Names and descriptions from this file are displayed in Customer Center and Provisioning Tags. See "[Customizing ERA Names and Descriptions for Client Applications](#)".

9. Enter values for the friends and family promotion for a specific customer who purchases a product or discount with the friends and family provisioning tag.

Important: If you do not add name-value pairs for friends and family ERAs, DAT_AccountBatch does not load the friends and family ERAs into the pipeline memory at startup.

Promotion is the term used for *ERA* in Customer Center. For example, for a telephony service, you would configure the promotion by entering phone numbers. For other services, you would enter the values relevant for the service, such as APN or email addresses.

If the ERA contains multiple lists, you configure each list in Customer Center.

10. To share ERA lists between accounts, create a profile sharing group.
11. (Optional) Create a profile sharing group to share the friends and family lists with other accounts or services.

Important: You can share service-level ERAs only.

When you configure a promotion in Customer Center, the values are stored in a profile. You can share that profile with services owned by accounts that own a product or discount with the same ERA.

Note: Typically, you would set up rating or discounting to use either the ERA label name with a model selector or the usage type. If you are using ERA labels and model selectors, you should comment out the IRL_UsageType iRule from the registry.

Sample Friends and Family Provisioning Tag

This is sample XML code for a provisioning tag that creates a friends and family ERA for an email service.

This code would be added to the `pin_config_provisioning_tags.xml` file:

```
<ProvisioningTag name="FRIENDS_FAMILY">
    <PermittedTypes>/service/email</PermittedTypes>
    <OpcodeList>
        <OpcodeName>PCM_OP_SUBSCRIPTION_PROVISION_ERA</OpcodeName>
        <OpcodeNumber>9066</OpcodeNumber>
        <OpcodeMode>0</OpcodeMode>
        <OpcodeParamsList>
            <OpcodeParamName>PIN_FLD_POID</OpcodeParamName>
            <OpcodeParamValue>0.0.0.0 /profile/serv_extrating
        -1</OpcodeParamValue>
        </OpcodeParamsList>
    </OpcodeList>
</ProvisioningTag>
```

```

<OpcodeParamsList>
  <OpcodeParamName>PIN_FLD_ACCOUNT_OBJ</OpcodeParamName>
  <OpcodeParamValue>$ACCOUNT$</OpcodeParamValue>
</OpcodeParamsList>

<OpcodeParamsList>
  <OpcodeParamName>PIN_FLD_FLAGS</OpcodeParamName>
  <OpcodeParamValue>0</OpcodeParamValue>
</OpcodeParamsList>

<OpcodeParamsList>
  <OpcodeParamName>PIN_FLD_SERVICE_OBJ</OpcodeParamName>
  <OpcodeParamValue>$SERVICE$</OpcodeParamValue>
</OpcodeParamsList>

<OpcodeParamsList>
  <OpcodeParamName>PIN_FLD_INHERITED_INFO.PIN_FLD_
NAME</OpcodeParamName>
  <OpcodeParamValue>FRIENDS_FAMILY</OpcodeParamValue>
</OpcodeParamsList>

<OpcodeParamsList>
  <OpcodeParamName>PIN_FLD_INHERITED_INFO.PIN_FLD_EXTRATING.PIN_
FLD_LABEL</OpcodeParamName>
  <OpcodeParamValue>MYFRIENDS</OpcodeParamValue>
</OpcodeParamsList>
</OpcodeList>

<OpcodeList>
  <OpcodeName>PCM_OP_SUBSCRIPTION_PROVISION_ERA</OpcodeName>
  <OpcodeNumber>9066</OpcodeNumber>
  <OpcodeMode>1</OpcodeMode>

  <OpcodeParamsList>
    <OpcodeParamName>PIN_FLD_POID</OpcodeParamName>
    <OpcodeParamValue>0.0.0.0 /profile/serv_extrating
-1</OpcodeParamValue>
  </OpcodeParamsList>

  <OpcodeParamsList>
    <OpcodeParamName>PIN_FLD_ACCOUNT_OBJ</OpcodeParamName>
    <OpcodeParamValue>$ACCOUNT$</OpcodeParamValue>
  </OpcodeParamsList>

  <OpcodeParamsList>
    <OpcodeParamName>PIN_FLD_FLAGS</OpcodeParamName>
    <OpcodeParamValue>1</OpcodeParamValue>
  </OpcodeParamsList>

  <OpcodeParamsList>
    <OpcodeParamName>PIN_FLD_SERVICE_OBJ</OpcodeParamName>
    <OpcodeParamValue>$SERVICE$</OpcodeParamValue>
  </OpcodeParamsList>

  <OpcodeParamsList>
    <OpcodeParamName>PIN_FLD_NAME</OpcodeParamName>
    <OpcodeParamValue>FRIENDS_FAMILY</OpcodeParamValue>
  </OpcodeParamsList>

```

```

        <OpcodeParamsList>
          <OpcodeParamName>PIN_FLD_INHERITED_INFO.PIN_FLD_EXTRATING.PIN_
FLD_LABEL</OpcodeParamName>
          <OpcodeParamValue>MYFRIENDS</OpcodeParamValue>
        </OpcodeParamsList>
      </OpcodeList>
</ProvisioningTag>

```

Note: The PCM_OP_SUBSCRIPTION_PROVISION_ERA opcode is specified twice, once with OpcodeMode and PIN_FLD_FLAGS set to 0 to run at purchase time and once with OpcodeMode and PIN_FLD_FLAGS set to 1 to run at cancel time. This opcode requires using either 0 or 1 for these values. See ["Specifying an Opcode in a Provisioning Tag to Create an ERA"](#).

To create an ERA with a second list, add two more <OpcodeList> blocks using the same opcode and the same values, except for a different value for PIN_FLD_INHERITED_INFO.PIN_FLD_EXTRATING.PIN_FLD_LABEL. Do the same if you want additional lists for the ERA.

Using ERAs with Multiple Lists

You can have multiple rates or discounts for an ERA with multiple ERA labels. An ERA label is an individual list of values, such as phone numbers or email addresses. A single ERA can have multiple ERA labels.

For example, a friends and family ERA can have separate lists for friends and family, each with a different rate. During rating or discounting, BRM chooses the correct rate based on the label name.

To set up rating and discounting for ERAs with multiple lists:

- For pipeline rating, see ["Using Model Selectors for ERAs"](#).
- For real-time rating, see ["Using Rate Plan Selectors for ERAs"](#).

Using Model Selectors for ERAs

You can use price or discount model selectors to define different rates or discounts for each ERA label in an ERA. Create model selector rules based on the PROFILE_LABEL_LIST EDR field. For each ERA label name, you can specify a different price or discount. During rating or discounting, Pipeline Manager matches a value in the EDR such as a phone number with the values in the ERA labels.

Model selectors also rank the price or discount models. If the EDR matches more than one ERA label, the first price model whose rule matches a label in the EDR is used.

In the following example, a friends and family ERA has two labels, MYFAMILY and MYFRIENDS. Each label is associated with a different price model through a price model selector, as shown in [Table 6-2](#):

Table 6-2 Example of Friends and Family ERA

Rank	DETAIL.PROFILE_LABEL_LIST	Price Model
1	*MYFAMILY*	1_cent_per_min
2	*MYFRIENDS*	2_cent_per_min

Tip: To use a discount with a discount model selector for a telco service, use the provisioning tag framework to create the tag with the ERA. You cannot use provisioning tags created with the Services Framework Manager with discounts. See "About Provisioning tags for telco services" in *BRM Telco Integration*.

For more information about defining price model selectors and discount model selectors, see Pricing Center Help.

Using Rate Plan Selectors for ERAs

You can define different rates for each ERA label in an ERA for real-time rating by using rate plan selectors. To do this, use the event attribute `EVENT.PIN_FLD_PROFILE_LABEL_LIST` in the rate plan selector matrix.

In the following example, a friends and family ERA has two labels, MYFAMILY and MYFRIENDS. Each label is associated with a different rate through a rate plan selector, as shown in [Table 6-3](#):

Table 6-3 Rate Plans for ERAs

Row	EVENT.PIN_FLD_PROFILE_LABEL_LIST	Rate Plan	Impact Category
1	*MYFAMILY*	1_cent_per_min	Default
2	*MYFRIENDS*	2_cent_per_min	Default

About Rating Based on Friends and Family ERAs

This section describes how BRM rates events with friends and family ERAs. For general information about ERAs, see "[About Extended Rating Attributes](#)". For information about working with multiple ERA lists, see "[Using ERAs with Multiple Lists](#)".

In both real-time rating and pipeline rating, the following occurs:

- If a predefined value in an event or EDR matches a value in an ERA, the names of the ERA and ERA labels are added to the event or EDR.
- If a rate plan selector or model selector has been defined based on the ERA label name, that selector is used to determine the rate. If an event has multiple labels, the rate plan selector or model selector also determines which label has priority.
- By default, friends and family ERAs are rated for telco services, but through customization they can be used and rated for any service.

For more information, see:

- [Real-Time Rating for Friends and Family ERAs](#)
- [Pipeline Rating for Friends and Family ERAs](#)

Real-Time Rating for Friends and Family ERAs

Friends and family ERAs can be rated based on either the ERA name, which is specified as the usage type, or the ERA label name.

In real-time rating, if a particular event attribute matches a value in a friends and family ERA, the ERA is added to the event's usage type field, and the ERA label is

added to the event's ERA label field. An event can match more than one friends and family label.

Using the ERA label name enables you to use different rates for different values within the same ERA. For example, you can use labels to charge different rates for calls to friends and calls to families by defining a rate plan selector. Using the usage type, you can charge a single rate for all friends and family calls.

Following is a summary of the real-time rating process for events with friend and family ERAs. The process is described in detail in "[Rating an Event Based on Extended Rating Attributes](#)". This section contains information specific to friends and family ERAs.

1. The PCM_OP_RATE_POL_PRE_RATING policy opcode calls the PCM_OP_RATE_POL_PROCESS_ERAS policy opcode, which in turn calls PCM_OP_RATE_GET_ERAS.
2. PCM_OP_RATE_GET_ERAS retrieves the valid ERAs for an event, including ERAs for the event's account, service, and shared profiles. The retrieved information includes the ERA name, label names, label values, and validity dates.
3. The PCM_OP_RATE_POL_PROCESS_ERAS policy opcode compares certain predefined event fields to the values for the ERA or ERAs valid for the event.

By default, PCM_OP_RATE_POL_PROCESS_ERAS is configured to work with a GSM service and to compare values in the PIN_FLD_CALLING_FROM and PIN_FLD_CALLED_TO fields.

PCM_OP_RATE_POL_PROCESS_ERAS can be easily configured to work with a GPRS service. In that case, it compares values in the PIN_FLD_ANI and PIN_FLD_DN fields.

Note: Using the PCM_OP_RATE_POL_PROCESS_ERAS policy opcode with services other than GSM and GPRS requires more extensive customization.

4. If a number in the PIN_FLD_CALLED_TO field matches a value in one or more of the friends and family ERA labels, PCM_OP_RATE_POL_PROCESS_ERAS adds the following to the event:
 - The ERA name to the PIN_FLD_USAGE_TYPE field. For a friends and family ERA, the name FF is added.
 - The ERA label name or names to the PIN_FLD_PROFILE_LABEL_LIST field. The names are added as they were defined in the ERA (for example, MYFRIENDS and MYFAMILY). Multiple names are separated by a comma, unless you specified a different separator by customizing the opcode.

The charge or discount for the event is then calculated based on the value of either the PIN_FLD_USAGE_TYPE or PIN_FLD_PROFILE_LABEL_LIST field.

If a rate plan selector was used, the rating engine uses it to find the correct rate. For a friends and family ERA, typically the rate plan selector would use the PIN_FLD_PROFILE_LABEL_LIST event field to map to different rates. For example, you could have a different rate for each ERA label.

Prioritizing Rates for ERAs in Real-Time Rating

The rate plan selector also ranks the rate plans. If the event matches more than one ERA label, the rating engine evaluates the rates in order of their rank in the rate plan selector and chooses the first rate that matches a label in the EDR.

The rate plan selector matches ERA labels within a list of multiple labels by using the *In List* matching type. This must be selected when you create the rate plan selector. The rating engine then treats PIN_FLD_PROFILE_LABEL_LIST or whichever fields you include in the rate plan selector as a list and parses it accordingly, using a separator you specify.

In the following example, the friends and family ERA has two labels, MYFAMILY and MYFRIENDS. Each label is associated with a different rate through a rate plan selector, as shown in [Table 6-4](#):

Table 6-4 Rate Plans and Multiple Labels

Row	EVENT.PIN_FLD_PROFILE_LABEL_LIST	Rate Plan	Impact Category
1	*MYFAMILY*	1_cent_per_min	Default
2	*MYFRIENDS*	2_cent_per_min	Default

If an event's PIN_FLD_PROFILE_LABEL_LIST is **MYFAMILY**, the rating engine chooses the **1_cent_per_min** rate plan. If PIN_FLD_PROFILE_LABEL_LIST contains the value **MYFAMILY, MYFRIENDS**, the rating engine again chooses the **1_cent_per_min** rate plan. Both rate plans are matched, but **1_cent_per_min** has higher priority.

Customizing Real-Time Rating for Friends and Family ERAs

You can customize the PCM_OP_RATE_POL_PROCESS_ERAS policy opcode to do the following:

- Add a service type, so you can rate ERAs for services other than telco services.

To do this, modify the `fm_rate_pol_proc_eras_set_era_name()` function.

To add a new service type, copy the existing code for SERVICE_TELEPHONY. To add a new service type, copy the code. In the copied code, change the service type, and change the PIN_FLD_CALLING_FROM and PIN_FLD_CALLED_TO fields to the appropriate fields for the new service type.

This is an excerpt from the default version of `fm_rate_pol_process_eras.c` showing the code you must copy and modify:

```
if(svc && !strncmp(svc, SERVICE_TELEPHONY, strlen(SERVICE_TELEPHONY))) {
    column_flistp = PIN_FLIST_ELEM_GET(e_flistp, PIN_FLD_CALL, rec_id, 1, ebufp);
    if(column_flistp) {
        a_num = PIN_FLIST_FLD_GET(column_flistp, PIN_FLD_ANI, 1, ebufp);
        b_num = PIN_FLIST_FLD_GET(column_flistp, PIN_FLD_DNIS, 1, ebufp);
    }
}
```

- Change the separator character used to separate multiple label names in the PIN_FLD_PROFILE_LABEL_LIST field. Comma is the default. You specify this character in the CM_OP_RATE_POL_PROCESS_ERAS policy opcode and in the rate plan selector in Pricing Center. Specify the same character in both places.

For information on how to specify the separator in the rate plan selector, see Pricing Center Help.

Pipeline Rating for Friends and Family ERAs

In pipeline rating, if ERA labels are used and a particular EDR field matches a value in a friends and family list, the ISC_ProfileAnalyzer iScript adds the ERA label to the EDR. An EDR can match more than one friends and family label. You set up a price or discount model selector to select a price or discount model based on the ERA label name. See ["Using Model Selectors for ERAs"](#).

If the IRL_UsageType iRule has been configured for ERAs, and a usage type set up in Pricing Center, the ERA name (not the label name) is added to the EDR's usage type field. See ["Configuring the IRL_UsageType iRule for ERAs"](#).

Typically, you would set up rating or discounting to use either the ERA label name with a model selector or the usage type. If you are using ERA labels and model selectors, you should comment out the IRL_UsageType iRule from the registry.

Following is a summary of the pipeline rating and discounting process for events with friend and family ERAs:

1. At startup, DAT_AccountBatch gets ERA and shared profile information from the BRM database and stores it in memory.
2. When a CDR is rated, DAT_AccountBatch determines the ERAs or profile sharing group that applies to the CDR, filters out the profiles specified in the **ProfilesNotLoadedIntoEdr** registry entry, and sends the remaining profile information to the FCT_Account module.
3. FCT_Account adds ERA and shared profile data to the EDR.
4. The ISC_ProfileLabel iScript calls the search method in DAT_AccountBatch to determine if any of the profiles that were filtered contains a value that matches the EDR field value. See ["Improving Pipeline Rating Performance for Events with ERAs"](#).

If there's a match, ISC_ProfileLabel does the following:

- a. Parses the ERA labels returned by the search method and populates them into the `DETAIL.PROFILE_LABEL_LIST` EDR field.
 - b. Populates the `DETAIL.USAGE_TYPE` EDR field based on the matching ERAs found.
5. For the profiles not specified in the `ProfilesNotLoadedIntoEdr` registry entry, the following occurs:
 - a. ISC_ProfileAnalyzer iscript analyzes the profiles and determines if any of the profiles contains a value that matches the EDR field value. If there is a match, the ERA label is added to the `DETAIL.PROFILE_LABEL_LIST` field in the EDR container.

By default, ISC_ProfileAnalyzer analyzes ERA profiles named `FRIENDS_FAMILY` for the service codes `TEL` (for GSM telephony) or `SMS`, by comparing the value of the `DETAIL.B_NUMBER` (called number) EDR field to the values in the ERA. You can customize ISC_ProfileAnalyzer to analyze other ERA profiles and service types. See ["Customizing Pipeline Rating for Friends and Family ERAs"](#).

6. The IRL_UsageType iRule adds the usage type to the `DETAIL.USAGE_TYPE` field in the EDR container if the conditions in the `IRL_UsageType.irl` file are met. For a friends and family ERA, the name added to `USAGETYPE` is `FF`.
7. For rating, FCT_MainRating finds the correct price model for calculating the charge as follows:

- If there is an ERA name in the USAGETYPE field, FCT_MainRating uses that ERA to rate the event.
 - If one or more ERA labels are listed in PROFILE_LABEL_LIST, FCT_MainRating chooses a price model using the rules in the price model selector.
8. For discounting, FCT_DiscountAnalysis finds the correct discount model to calculate the discount, using the values in the PROFILE_LABEL_LIST EDR field and the rules in the discount model selector.

The FCT_Discount module applies the discount after FCT_DiscountAnalysis determines the discount model to use.

Prioritizing Rates and Discounts for ERAs in Pipeline Rating

The price or discount model selector ranks the price or discount models. If the EDR matches more than one ERA label, Pipeline Manager evaluates the price or discount models in order of ranking and chooses the first price or discount model whose rule matches a label in the EDR.

In the following example, the friends and family ERA has two labels, MYFAMILY and MYFRIENDS. Each label is associated with a different price model through a price model selector, as shown in [Table 6-5](#):

Table 6-5 Price Model Selector

Rank	DETAIL.PROFILE_LABEL_LIST	Price Model
1	*MYFAMILY*	1_cent_per_min
2	*MYFRIENDS*	2_cent_per_min

If an EDR's DETAIL.PROFILE_LABEL_LIST is **MYFAMILY**, the rating module chooses the **1_cent_per_min** price model. If DETAIL.PROFILE_LABEL_LIST contains the value **MYFAMILY**, **MYFRIENDS**, FCT_MainRating again chooses the **1_cent_per_min** price model. Both price models are matched, but **1_cent_per_min** has higher priority.

Improving Pipeline Rating Performance for Events with ERAs

When Pipeline Manager processes a call detail record (CDR), all the ERA profiles (owned or shared) that apply to the CDR are loaded into the event data record (EDR) container. However, some accounts may own a large number of ERA profiles or share a large number of ERA profiles in a profile sharing group. The populating of a large number of profiles into the EDR container may result in performance degradation. For example, when a Friends and Family profile sharing group contains a large number of ERA profiles that are shared with accounts in a hierarchy, all the shared profiles in the group are populated into the EDR container during CDR processing of the member accounts. In this example, the population of all the shared profiles into the EDR containers for *each* account in the hierarchy results in performance degradation.

When you have accounts that own or share a large number of ERA profiles, you can use the DAT_AccountBatch registry entry, **ProfilesNotLoadedIntoEdr**, to specify the profiles to be filtered. When a CDR is rated, if the value in the EDR field matches a value in one or more ERA profiles in the **ProfilesNotLoadedIntoEdr** registry entry, the ISC_ProfileLabel iScript loads only the ERA labels into the EDR container.

To configure batch pipeline rating to filter ERA profiles:

1. Set the DAT_AccountBatch module registry entry **ProfilesNotLoadedIntoEdr** to the profiles that should be filtered. You can specify any number of profiles separated by a comma.

For example:

```
ProfilesNotLoadedIntoEdr {FRIENDS_FAMILY, DISCOUNTBUNDLE}
```

2. Configure the ISC_ProfileLabel iScript. See "ISC_ProfileLabel" in *BRM Configuring Pipeline Rating and Discounting*.

Customizing Pipeline Rating for Friends and Family ERAs

You customize pipeline rating for friends and family ERAs by customizing the ISC_ProfileAnalyzer and ISC_ProfileLabel iScripts.

By default, ISC_ProfileAnalyzer and ISC_ProfileLabel analyzes ERAs named FRIENDS_FAMILY for the service codes TEL (for GSM telephony) or SMS, comparing the value of the DETAIL.B_NUMBER (called number) field in the EDR to the ERA values.

To use ISC_ProfileAnalyzer and ISC_ProfileLabel with other services and ERAs, add an ELSE IF block to the onDetailEdr function. The default version of ISC_ProfileAnalyzer includes this example for adding a GPRS service:

```
else
if (edrServiceType == "GPRS")
{
    compareString = edrString(DETAIL.ASS_GPRS_EXT.APN_ADDRESS,0);
    compProfName = "FRIENDS_FAMILY";
}
}
```

This block includes the entries shown in [Table 6–6](#):

Table 6–6 Entries for GPRS Service

String	Description
edrServiceType	The service code.
compareString	The EDR field to compare to the ERA.
compProfName	The name of the ERA to analyze to compare to the EDR field.

Then you edit this information in the registry:

- Service type: Change TEL to the new service code.
- Profile name: Change FRIENDS_FAMILY to a different ERA name if needed.
- Separator: Change a comma to another character to separate multiple label names in EDRs, if needed.

Customizing ERA Names and Descriptions for Client Applications

You customize ERA names and descriptions that are displayed on the **Promotion** tab in Customer Center by editing the `era_descr.en_US` file in the `BRM_Home/sys/msg/eradescr` directory. You can localize this file.

Note: For ERAs defined in telco provisioning tags, the names and descriptions are also used in the Provisioning Tags application.

This file includes entries for ERA names and descriptions that are identified by ID numbers. Each name and description must have a unique ID number.

You define the ID numbers differently depending on how the ERAs are defined:

- For ERAs defined in telco provisioning tags, the ID numbers must match the ID numbers used in `pin_telco_tags_gsm` or other `pin_telco_tags` files that you edit when configuring ERAs.

For example, this is the entry for the SPECIAL_DAY account-level ERA in the `pin_telco_tags_gsm` file:

```
account_era "SPECIAL_DAY" 2 3
```

- For ERAs defined using the provisioning tag framework, you include each ID in the `customized.properties` file in the Customer Center SDK. You also must include the name of the ERA in the provisioning tag definition.

See ["Creating Names and Descriptions for ERAs Defined in the Provisioning Tag Framework"](#).

The following example shows the name and description of the Special Day ERA in the `era_descr.en_US` file:

```
ID = 2 ;
VERSION = 1 ;
STRING = "Special Day" ;
```

```
ID = 3 ;
VERSION = 1 ;
STRING = "This option gives discounts on calls made on your birthday. An account can have only one Birthday. To give this promotion, enter the word BIRTHDAY in the Name field, and enter the customer's birthday in the Value field." ;
```

When you add or change ERA names and descriptions, you might also need to provide more detailed information for your CSRs. For example, you could provide a list of valid entries for the Customer Type ERA and instructions on when to use each entry.

After you customize the file, use the `load_localized_strings` utility to load the contents of the `era_descr.en_US` file, or a localized version of the file, into the `/strings` object.

To load the contents of `era_descr.en_US`, use this command:

```
load_localized_strings era_descr.en_US
```

For information on loading the `era_descr.locale` file, see "Loading Localized or Customized Strings" in *BRM Developer's Guide*. For information on creating new strings for this file, see "Creating New Strings and Customizing Existing Strings" in *BRM Developer's Guide*.

Note:

- ERA label names are not included in the `era_descr.en_US` file and cannot be localized.
 - Default ERA names and descriptions are loaded when you install a service manager. You must load them again only if you customize them.
 - If you're loading a localized version of this file, use the correct file extension for your locale.
-
-

Creating Names and Descriptions for ERAs Defined in the Provisioning Tag Framework

If you define an ERA using the provisioning tag framework, use the following procedure to create a name and description that appears in Customer Center:

1. Add the name and description to the `era_descr.en_US` file and to any other localized versions of the ERA description file you are using.

For information, see "[Customizing ERA Names and Descriptions for Client Applications](#)".

Important:

- You must use a unique ID number for each name and description you add.
 - You must use the same ID number for the same ERA in each localized version of the ERA description file.
-
-

2. Load the names and descriptions using the `load_localized_strings` utility. See "[Customizing ERA Names and Descriptions for Client Applications](#)".
3. Configure the ERA in the provisioning tag framework as follows:
 - Configure the provisioning tag to call the `PCM_OP_SUBSCRIPTION_PROVISION_ERA` opcode during purchase and cancel modes.
 - Specify the ERA name in the `PIN_FLD_NAME` parameter for the opcode. For example, for a friends and family ERA, specify `FRIENDS_FAMILY` in the provisioning tags configuration file.
 - Set the `PIN_FLD_STR_VAL` parameter to **12, 13** so that the profile name and profile description are localized and are stored in the `/string` object.

For an example of a provisioning tag configuration for an ERA, see "[Sample Friends and Family Provisioning Tag](#)". For information on using the provisioning tag framework, see "[Using the Provisioning Tag Framework](#)".

Rating Implementation and Customization

This chapter provides information on implementing and extending the default Oracle Communications Billing and Revenue Management (BRM) rating functionality.

For general information on rating, see the following topics:

- [About Creating a Price List](#)
- "About Pipeline Rating" in *BRM Configuring Pipeline Rating and Discounting*

How Rating Works

The PCM_OP_ACT_USAGE opcode is the main BRM rating opcode. In addition, subscription opcodes, such as PCM_OP_SUBSCRIPTION_CYCLE_FORWARD, are used for applying fees.

How BRM Rates and Records Usage Events

BRM uses PCM_OP_ACT_USAGE to rate usage events and record them in the BRM database. For example, this opcode is called when an event is generated for a user login session.

Important: Do not call PCM_OP_ACT_USAGE directly.

PCM_OP_ACT_USAGE takes as input an **/account** object POID (Portal Object ID) and a type-only POID that specifies the type of event being recorded.

The PIN_FLD_PRODUCTS array may be included in the input flist to supply or override the list of products that the account owns.

PCM_OP_ACT_USAGE performs the following operations:

1. **Determines whether the event is pre-rated.** If the event is pre-rated, the opcode does not rate the event. However, to support taxation for the following adjustments, disputes, and settlements, this opcode performs additional filtering for pre-rated events:
 - Adjustments at the account level, subscription service level, member service level, and item level
 - Item-level disputes
 - Item-level settlements

This additional filtering consists of checking the event type and performing one of the following activities, as applicable:

- **Adjustments at the account level, subscription service level, member service level:** Gets the adjustment amount and then requests deferred or immediate tax calculation for this amount.
- **Item-level adjustments, disputes, or settlements:** Evaluates the adjustment, dispute, or settlement amount for each event in the bill item and determines the proportionate amount for taxation. It then requests deferred or immediate tax calculation for this amount.

BRM chooses whether to use deferred or immediate taxation for adjustments, disputes, and settlements based on the **tax_now** entry in the Connection Manager (CM) configuration file (**pin.conf**).

For event-level disputes and settlements, PCM_OP_ACT_USAGE also processes notification events that include all the account and balance impact information required to reserve disputed resources and to release them upon settlement. For more information, see "Configuring Adjustments, Disputes, and Settlements" in *BRM Managing Accounts Receivable*.

2. **Adjusts available balances to include reservations.** When PCM_OP_ACT_USAGE is called to *reauthorize* an ongoing prepaid session, an array of the resources currently reserved for the session (the PIN_FLD_RESERVATION_LIST array) is included in the input flist. PCM_OP_ACT_USAGE gets the account balances from which these reservations were made. It then temporarily adds the amount currently reserved for the session to the remaining *unreserved* amount in each retrieved balance. The sum of these two amounts represents the total amount of the resource available to the ongoing session. This amount is used to rate the reauthorization request.

For example, customer X has a resource balance of 70 free minutes. Currently, 30 of the minutes are reserved for session A, 30 are reserved for session B, and 10 are unreserved. BRM receives a request to extend session A for 30 minutes. Before the request is rated to determine whether customer X's account has sufficient resources for it, PCM_OP_ACT_USAGE adds the free minutes currently reserved for session A (30) to the account's unreserved free minutes (10) to get the total free minutes available to session A (30 + 10 = 40). In this case, the total free minutes available to session A (40) does not cover the reauthorization request, which is for 60 minutes (30 for the initial authorization + 30 for the extension).

3. **Determines whether the event is discountable.** PCM_OP_ACT_USAGE determines whether the event is discountable.

Note: Discountable events include events to which you can apply product discounts, charge sharing, discount sharing, loyalty points, free minutes, and so forth.

By default, the rating opcodes perform a credit limit check when rating an event. In CALC_ONLY mode, the credit limit check includes the effects of discounts. Because discounts can reduce the balance impact of an event that might otherwise exceed an account's credit limit, the credit limit check for discountable events is deferred to a real-time discounting pipeline.

To determine whether an event is discountable, PCM_OP_ACT_USAGE checks the usage maps of all the discounts in your system. If at least one map contains the event type of the event being rated, the event is considered discountable.

Note: PCM_OP_ACT_USAGE evaluates discount usage maps using the data that exists as of the most recent restart of the CM. Instances of event types that are newly added to discount usage maps after the most recent CM restart are therefore not found to be discountable. For example, if you add a new real-time usage event type to an existing discount that does not already include a real-time usage event, events of that type are not found to be discountable. You must restart the CM to make these newly added event types discountable.

If the event is discountable, PCM_OP_ACT_USAGE retrieves all candidate discounts associated with the customer's account. If one or more candidate discounts are returned, the PCM_OP_RATE_EVENT opcode is called with the PCM_OPFLG_RATE_WITH_DISCOUNTING flag set to **True**. This flag turns off the credit limit check performed by the real-time rating opcodes and defers it to the real-time discounting pipeline.

See "[Rating Events](#)" for more information about PCM_OP_RATE_EVENT.

4. **Determines how much to charge for the event.** If the calling opcode does not specify a rate, PCM_OP_ACT_USAGE does the following:
 - Reads the `/config/rum` object to generate a ratable usage metric (RUM) candidate for rating the event.
 - Calls PCM_OP_RATE_EVENT to rate the event. See "[FM Rate Opcodes Called by PCM_OP_ACT_USAGE](#)".
 - Calls the PCM_OP_RATE_POL_POST_RATING policy opcode to apply any modifications to the `/event/activity` object after rating the event. See "[Modifying Rated Events](#)".
5. **Applies the event's balance impact.** PCM_OP_ACT_USAGE applies the balance impact of the event to the account balances as follows:
 - If the account has resources set aside in reservations, the resources are consumed from the reservations.
 - If the account does not have reserved resources, the event balance impact is applied to a balance group in the following order based on what is provided in the input flist:
 - If a bill unit is specified in the input flist, the event balance impact is applied to the default balance group of the bill unit.
 - If a bill unit is not specified, the event balance impact is applied to the specified balance group.
 - If a balance group is not specified, the event balance impact is applied to the balance group of the specified service.
 - Otherwise, the event balance impact is applied to the default account-level balance group.

Note: If the PCM_OPFLG_CALC_ONLY or PIN_ACT_CALC_MAX flag is used, the account balance is not updated.

6. **Records the event.** If no balance impact is associated with the event, PCM_OP_ACT_USAGE checks the Event Record Map configuration object (`/config/event_`

record_map cached at CM startup time) against the input event type. If the input event type is configured not to be recorded in BRM, the event is not recorded. Otherwise, the event is recorded.

PCM_OP_ACT_USAGE performs these tasks:

- Updates the event fields using information from the specified **/account** object. PCM_OP_ACT_USAGE associates the **/item** object POID with the event and sets the PIN_FLD_CURRENCY, **/payinfo**, **/bill** object POID, and general ledger (G/L) segment ID.

Note: PCM_OP_ACT_USAGE does not record the PIN_FLD_EXTENDED_INFO substruct field in the event object. To record the PIN_FLD_EXTENDED_INFO substruct in the event object for customizations, modify the PCM_OP_RATE_POL_POST_RATING policy opcode.

- Obtains information about event fields to be cached for event searches and invoicing.
- Retrieves a list of remittance accounts that must be remitted for the event.

Note: BRM caches remittance specifications. The PCM_OP_REMIT_GET_PROVIDER opcode is called only when the remittance specification cache is non-null.

- Checks whether the event is included in your system's event notification list. If it is, calls the opcodes associated with the event in the list.

Specifying the Rating Mode

You control how PCM_OP_ACT_USAGE rates and records usage events by using the following flags:

- **PCM_OPFLG_CALC_ONLY**

When this flag is passed in the opcode call, PCM_OP_ACT_USAGE calculates the amount to charge, factoring in all applicable discounts, but does not apply the amount to the account balance.

- **PIN_ACT_CALC_MAX**

When PIN_FLD_FLAGS is set to PIN_ACT_CALC_MAX, PCM_OP_ACT_USAGE calculates the maximum quantity that can be consumed based on the event owner's current account balance or reserved amount but does not apply the amount to the account balance.

- **PIN_RATE_FLG_RATE_ONLY**

When PIN_FLD_FLAGS in the PIN_FLD_BAL_IMPACTS array is set to PIN_RATE_FLG_RATE_ONLY, PCM_OP_ACT_USAGE uses rated, pre-rated, and tax balance impact types to rate the event.

Note: When PIN_RATE_FLG_RATE_ONLY is set, PCM_OP_ACT_USAGE returns the event's PIN_FLD_BAL_IMPACT array in the output flist.

- **PIN_RATE_FLG_RERATE**

When the PIN_FLD_FLAGS field in the PIN_FLD_BAL_IMPACTS array is set to PIN_RATE_FLG_RERATE, PCM_OP_ACT_USAGE uses rerated tax balance impact types to rate the event.

Note: When PIN_RATE_FLG_RERATE is set, PCM_OP_ACT_USAGE returns the event's PIN_FLD_BAL_IMPACT array in the output list.

- **PIN_RATE_FLG_OVERRIDE_CREDIT_LIMIT**

When PIN_FLD_FLAGS in the PIN_FLD_BAL_IMPACTS array is set to PIN_RATE_FLG_OVERRIDE_CREDIT_LIMIT, PCM_OP_ACT_USAGE does not check the credit limit.

FM Rate Opcodes Called by PCM_OP_ACT_USAGE

To rate events, PCM_OP_ACT_USAGE calls the following opcodes:

- PCM_OP_RATE_EVENT. See ["Rating Events"](#).
- PCM_OP_RATE_GET_PRODLIST. See ["Retrieving Product Lists"](#).

Rating Events

To rate an event, PCM_OP_RATE_EVENT does the following:

- When the optional time-stamp field PIN_FLD_WHEN_T is present in the PCM_OP_RATE_EVENT input list, PCM_OP_RATE_EVENT searches for the price list that is valid at the time specified in the field. It uses the price list to rate the event.
- PCM_OP_RATE_EVENT sets the optional PIN_FLD_VALID_TO field for CALC_MAX and CALC_ONLY operations. This field is used to limit prepaid product authorizations to a specific time period for non-duration RUMs.
- If a product or discount starts on first usage and its validity period has not been set, selects the product or discount as a candidate for rating if the purchase time is equal to or earlier than the event time. The opcode returns the first-usage products that are used to rate the event in the PIN_FLD_FIRST_USAGE_PRODUCTS array in the output list.
- If rating impacts a balance whose validity period has not yet been set (such as non-currency balances that start on first usage or relative to when they are granted), calculates the resource balance validity period. If resource validity end time is restricted to the granting product's or discount's end time, this opcode compares the calculated end time with the product's or discount's end time and returns the earlier of the two in the output list.
- Locates any rollover objects for events and returns details of the rollover object to the calling opcode.
- Determines whether it should perform a credit limit check. PCM_OP_RATE_EVENT performs a credit check unless:
 - The event is discountable or involves multiple RUMs. In CALC_ONLY mode, the credit limit check is deferred to the real-time pipeline.
 - The event being rated includes the PIN_RATE_FLG_NO_CREDIT_LIMIT_CHECK flag, the applicable rate is a credit limit override rate, or the event is a refund event. (The PIN_RATE_FLG_NO_CREDIT_LIMIT_CHECK flag is set if

the event associated with the plan does not use a credit limit. You specify to not use a credit limit in Pricing Center when you set up a plan.)

PCM_OP_RATE_EVENT sets a value in the PIN_FLD_CHECK_CREDIT_LIMIT field in its output flist to indicate whether a credit limit check should occur, where it should occur, and how the real-time discounting pipeline should handle the event if the credit limit occurs there. Flag values can be summed.

- PIN_RATE_CREDIT_LIMIT_NEEDED (0x1000) indicates that a credit limit check should be performed.
 - PIN_RATE_CREDIT_LIMIT_LOCATION_CRE (0x100) indicates that the credit limit check is handled by PCM_OP_RATE_EVENT.
 - PIN_RATE_CREDIT_LIMIT_LOCATION_RTP (0x200) indicates that the credit limit check is deferred to the real-time discounting pipeline.
 - PIN_RATE_NO_CREDIT_LIMIT_DISCOUNT_ONLY (0x0): Indicates, for events that are processed by the real-time discounting pipeline, that no credit limit check should occur. This flag is used for events sent to the pipeline for discounting only, typically when rating in normal, as opposed to CALC_ONLY, mode.
 - PIN_RATE_CREDIT_LIMIT_CHECK (0x1): Indicates, for events that are processed by the real-time discounting pipeline, that a credit limit check should be performed by the FCT_CreditLimitCheck module. This flag is typically used for prepaid authorization. If the full amount cannot be authorized, FCT_CreditLimitCheck determines the maximum amount that can be authorized.
- If the event qualifies for a discount, sends the event to a real-time discounting pipeline.
 - Calls the PCM_OP_RATE_TAX_CALC opcode to calculate taxes.
 - When rating cycle forward events, calculates fees and refunds based on scale values, proration settings, and rates. The opcode uses the values of PIN_FLD_SCALE and PIN_FLD_ORIGINAL_SCALE passed to it by Subscription Manager to determine the scale to apply. These values are set differently depending on whether they are used for applying cycle fees or refunding them.

The opcode calculates the cycle fee or refund based on the following formula:

PIN_FLD_SCALE * PIN_FLD_ORIGINAL_SCALE * Rate

The opcode retains the value of PIN_FLD_ORIGINAL_SCALE that is passed to it, but it sometimes changes the value of PIN_FLD_SCALE depending on the proration setting.

To calculate cycle fees, the opcode sets PIN_FLD_SCALE to the following values:

- When purchase proration is set to **Do not charge for this cycle**, PIN_FLD_SCALE is set to **0**.
- When purchase proration is set to **Charge for the entire cycle**, PIN_FLD_SCALE is set to **1**.
- When purchase proration is set to **Calculate the charge based on the amount used**, PIN_FLD_SCALE retains the value passed to the opcode by Subscription Manager.

To calculate cycle fee refunds, the opcode sets PIN_FLD_SCALE to the following values:

- When cancel proration is set to **Do not charge for this cycle**, PIN_FLD_SCALE is set to **1**.
- When cancel proration is set to **Charge for the entire cycle**, PIN_FLD_SCALE is set to **0**.
- When cancel proration is set to **Calculate the charge based on the amount used**, PIN_FLD_SCALE retains the value passed to the opcode by Subscription Manager.

The opcode includes PIN_FLD_SCALE and PIN_FLD_ORIGINAL_SCALE in its output flist. These fields are stored in the `/event/billing/product/fee/cycle` object.

Retrieving Product Lists

To get the list of products for an event, PCM_OP_ACT_USAGE calls the PCM_OP_RATE_GET_PRODLIST opcode. This opcode gets a list of purchased products (`/purchased_product` objects) for an account based on the combination of service and event type in its input flist. It returns a list of base products and valid customized products.

In its initial search for products, the opcode uses the event object and optional service object in the flist to determine which products it retrieves:

- If the input flist does not include a service object POID, the opcode finds all purchased products for the account.
- If a service is included in the input flist, the opcode selects products that apply to the specified service and the event type in the flist. For example, if the event type is `/event/session/telco/gsm` and the service is `/service/telco/gsm/telephony`, the opcode finds products that apply to GSM usage.

After retrieving the list of products, the opcode checks if any of the products are overridden by a customized product that is currently valid. If a product is overridden by a valid customized product, it is removed from the list and not returned by the opcode. The valid customized product is included in the list instead.

If a product is retrieved and its validity period is not yet initialized, information about the product's purchase, cycle, and usage start and end times is returned in the `purchase`, `cycle`, and `usage *_START_DETAILS` and `*_END_DETAILS` fields respectively.

Generating Ratable Usage Metrics

Use the PCM_OP_ACT_GET_CANDIDATE_RUMS opcode to generate a ratable usage metric (RUM) that can be used to rate the event. A candidate RUM specifies the name, quantity, and unit of measurement for each resource used in the event. For example, an IP fax event's RUM might include the information in [Table 7-1](#):

Table 7-1 *Generating Ratable Usage Metrics*

Name	Quantity	Unit
duration	2	minutes
page_count	10	pages
byte_count	9568	bytes

PCM_OP_ACT_GET_CANDIDATE_RUMS performs the following operations:

1. Calls the PCM_OP_ACT_POL_SPEC_CANDIDATE_RUMS policy opcode to calculate RUMs. See "[Customizing How to Calculate RUMs](#)".
2. Reads the `/config/rum` object to retrieve the list of valid RUMs for the specified event type.
3. Combines the configuration data with the event data to produce a RUM candidate for rating the event.
4. Returns the following data about each RUM:
 - RUM name
 - Quantity to rate
 - Unit (seconds, minutes, bytes)

About Calculating the Maximum Available Usage

BRM uses the PCM_OP_ACT_CALC_MAX_USAGE opcode to calculate the maximum available usage based on one of the following:

- The account's credit limit and current account balance.
- The amount reserved for the session.

For example, this opcode is called by the PCM_OP_ACT_AUTHORIZE opcode to calculate the maximum possible duration of a call to prevent prepaid customers from exceeding their current account balance.

PCM_OP_ACT_CALC_MAX_USAGE performs the following operations:

1. Specifies a huge quantity for the event RUM.
2. Calls PCM_OP_ACT_USAGE with PIN_FLD_FLAG set to PIN_ACT_CALC_MAX to rate and return the maximum quantity available based on the current balance or the reservation amount. See "[How BRM Rates and Records Usage Events](#)".
3. Returns PIN_FLD_QUANTITY set to one of the following:
 - The maximum usage allowed based on the user's available balance and credit limit or reserved amount.
 - **-1** to indicate unlimited usage. This value is returned only when *all* of the account's available resources can be consumed.

Note: By default, for a duration-type RUM, PCM_OP_ACT_CALC_MAX_USAGE limits the maximum duration to 24 hours. For example, if a user's available balance enables the user to make a call longer than 24 hours, this opcode returns **-1**. To configure the duration maximum, set the `fm_act_max_qty_for_duration` entry to a value greater than 24 (hours) in the CM configuration file.

This opcode propagates any errors returned from PCM_OP_ACT_USAGE.

About Currency Conversion

During rating, currency conversion is handled by the following opcodes:

- The PCM_OP_BILL_CURRENCY_CONVERT_AMOUNTS opcode converts the currency according to the conversion rate defined in the

`/config/currency/conversionrates` object. You can set multiple time periods for conversion rates.

This opcode fails if the specified time is not within the time range or if the source or destination currency is invalid. The `ERROR_NOT_FOUND` error code is returned.

- The `PCM_OP_BILL_CURRENCY_QUERY_CONVERSION_RATES` opcode retrieves the conversion rates from the `/config/currency/conversionrates` object. This opcode is called by the `PCM_OP_BILL_CONVERT_AMOUNTS` opcode.

This opcode fails if no conversion rate is specified between the source and destination currency types. The `ERROR_NOT_FOUND` error code is returned.

About Applying Cycle Forward Fees

To apply a cycle forward fee to a balance group, use the `PCM_OP_SUBSCRIPTION_CYCLE_FORWARD` opcode.

This opcode is called to charge or refund cycle forward fees (for example, when a product or discount is purchased, canceled, activated, or inactivated).

`PCM_OP_SUBSCRIPTION_CYCLE_FORWARD` performs the following operations:

1. When a product or discount is *purchased* or *activated* during the cycle, determines scale values that are used by real-time rating to prorate the cycle forward fee amount to be charged.
2. When a product or discount is *canceled* or *inactivated* during the cycle, determines scale values that are used by real-time rating to prorate the cycle forward fee amount to be refunded.
3. Uses customized products when valid to calculate fees or refunds. If a customized product is valid for only part of a cycle, it contributes toward the total charge or refund based on the length of its validity.
4. When a rate change is scheduled for the *next* or *current* cycle, `PCM_OP_SUBSCRIPTION_CYCLE_FORWARD` gets a list of rates and the period in which they are applicable and calculates the correct charges.
5. Calculates and sets `CYCLE_FEE_START_T` and `CYCLE_FEE_END_T` to the next period, if applicable.
6. After all the products and discounts and their cycle forward rates are determined, it sends the information to `PCM_OP_ACT_USAGE` to rate and apply the charges.

Note: If the product or discount starts on first usage (when the customer first uses the product or discount) and its validity period has not yet been set, `PCM_OP_SUBSCRIPTION_CYCLE_FORWARD` does not call `PCM_OP_ACT_USAGE`, and cycle fees are not applied.

7. For auditing purposes, it creates `/event/billing/product/fee/cycle/cycle_forward_type` objects, where *type* is the frequency of the cycle forward charge (for example, daily, weekly, monthly, bimonthly, quarterly, semiannual, or annual).
8. If successful, it returns the POIDs of the `/account` object and the `/event/billing/product/fee/cycle/cycle_forward_type` event.

About Applying Cycle Arrears Fees

To apply a cycle arrears fee, use the PCM_OP_SUBSCRIPTION_CYCLE_ARREARS opcode.

Note: Cycle arrears fees are applied only for a single month.

PCM_OP_SUBSCRIPTION_CYCLE_ARREARS performs the following operations:

1. When products or discounts with cycle fees are canceled or inactivated during a cycle, calculates the scale to prorate the cycle arrears fee amount to be charged.
2. When a rate change occurred in the *previous* or *next* cycle, gets a list of rates and the period in which they are applicable and calculates the correct charges.
3. Uses customized products when valid to calculate fees or refunds. If a customized product is valid for only part of a cycle, it contributes a portion of the total charge or refund based on the length of its validity.
4. After all the product and discount cycle arrears rates are determined, sends the information to PCM_OP_ACT_USAGE to rate and apply the charges.

Note: If the product or discount starts on first usage (when the customer first uses the product or discount) and its validity period has not yet been set, PCM_OP_SUBSCRIPTION_CYCLE_ARREARS does not call PCM_OP_ACT_USAGE, and cycle fees are not applied.

5. Creates the `/event/billing/product/fee/cycle/cycle_arrears` event for auditing purposes.
6. If successful, returns the POIDs of the `/account` object and the `/event/billing/product/fee/cycle/cycle_arrears` event.

Customizing the Cycle Interval for Products

To customize the time interval for applying cycle forward and cycle arrears fees for a specified product, use the PCM_OP_SUBSCRIPTION_POL_SPEC_CYCLE_FEE_INTERVAL policy opcode.

This policy opcode is called by PCM_OP_SUBSCRIPTION_CYCLE_FORWARD and PCM_OP_SUBSCRIPTION_CYCLE_ARREARS. The type of cycle event is passed in the PIN_FLD_SCOPE field in the input flist.

By default, this policy opcode is an empty hook to facilitate customization of the cycle forward and cycle arrears start (CHARGED_FROM_T) and end dates (CHARGED_TO_T) for a specific product. The start and end dates provided are used by rating opcodes to calculate the scale to determine the cycle fee amount to charge or refund.

The PIN_FLD_SCALE value in the input and output flists is the original charge scale and is used only to calculate the refund scale.

For example, if a product is purchased on April 1, 2009, with a monthly cycle forward fee of \$30 and the purchase, usage, and cycle end dates are set to April 20, 2009, the cycle fee amount is based on the scale for the period April 1, 2009, to April 20, 2009 (20 days) divided by the unit interval from April 1, 2009, to April 30, 2009 (30 days). The cycle fee charged is $20/30 * \$30$, or \$20.

If the product is canceled on April 15, 2009, the refund amount is based on the scale for the period April 15, 2009, to April 20, 2009 (5 days) divided by the unit interval from April 1, 2009, to April 20, 2009 (20 days). Because the refund amount is refunded from the charged amount (\$20), the refund is $5/20 * (20/30 * \$30)$, or \$5. Here, the scale value $20/30$ is the original charge scale or the period the product was valid during the cycle.

To change the scale (for example if you do not want to refund the full amount), change the start and end dates. The refund scale is calculated based on the dates that you provide.

About Restricting the End Time of Granted Resources That Start on First Usage

Note: This is an optional task that you perform only when you configure the validity periods of granted resources to start on first usage. For more information, see ["About Balance Impacts That Become Valid on First Usage"](#).

You can configure BRM to automatically restrict the validity period of granted resources to end no later than the end time of the product or discount that grants the resource.

When you configure resource validity to start on first usage and end on a date relative to the start date (when first usage occurs), you cannot know the actual end date when setting up the rate plan. Restricting the resource end time to the product or discount end time ensures that the resource cannot continue to be consumed after the product or discount expires.

Note: When a product or discount is *canceled*, the validity period end time of resources granted by that product or discount is set to the time of the cancellation.

When you restrict resource validity end time, BRM sets the end time to the product or discount end time at the time of the grant. When the customer consumes the granted resource for the first time, the relative end time is calculated:

- If the calculated end time is *later* than the granting product or discount end time, the resource validity period uses the product or discount end time.
- If the calculated end time is *earlier* than the granting product or discount end time, the resource validity period uses the calculated end time.

Important: If you restrict resource validity end time, you must do so for both real-time rating and pipeline rating.

To restrict resource validity end time, see ["Configuring Real-Time Rating to Restrict Resource Validity End Time"](#).

Configuring Real-Time Rating to Restrict Resource Validity End Time

In real-time rating, you restrict resource validity end time to the end time of the product or discount that grants the resource by modifying a field in the `multi_bal` instance of the `/config/business_params` object.

You modify the `/config/business_params` object by using the `pin_bus_params` utility.

To restrict resource validity end times to product or discount end times:

1. Run the following command, which creates an editable XML file from the **multi_bal** instance of the **/config/business_params** object:

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

This command creates the XML file named **bus_params_multi_bal.xml.out** in your working directory. If you do not want this file in your working directory, specify the path as part of the file name.

2. Search the XML file for following line:

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

3. Change **FALSE** to **TRUE**.

Caution: BRM uses the XML in this file to overwrite the existing **multi_bal** instance of the **/config/business_params** object. If you delete or modify any other parameters in the file, these changes affect the associated aspects of the BRM multi-balance configuration.

4. Save the file and rename it from **bus_params_multi_bal.xml.out** to **bus_params_multi_bal.xml**.
5. Run the following command, which loads the change into the **/config/business_params** object:

```
pin_bus_params bus_params_multi_bal.xml
```

Run this command from the *BRM_Home/sys/data/config* directory (where *BRM_Home* is the directory in which BRM is installed), which includes support files used by the utility.

6. Read the object with the **testnap** utility or Object Browser to verify that all fields are correct.
7. Stop and restart the CM.
8. (Multischema systems only) Run the **pin_multidb** script with the **-R CONFIG** parameter. For more information, see "pin_multidb" in *BRM System Administrator's Guide*.

About Applying Folds

For background information about folds, see "[About Fold Events](#)".

Use the **PCM_OP_SUBSCRIPTION_CYCLE_FOLD** opcode to apply balance impacts for fold events.

PCM_OP_SUBSCRIPTION_CYCLE_FOLD performs the following operations:

1. Calls the **PCM_OP_SUBSCRIPTION_POL_SPEC_FOLD** policy opcode to get the order in which to fold the account's resources. By default, folds are applied in ascending order of the resource IDs.
2. Checks the value of the resource's **PIN_FLD_APPLY_MODE** field in the **/config/beid** object to see whether the resource is specified to be rolled over. For more information, see "[Specifying Which Resources to Fold](#)".

3. Applies folds for all of the account's balance groups and also for each resource in the balance group that is specified to be rolled over. However, if a resource ID is specified in the input flist, only that resource is folded.
4. After applying the balance impacts, creates `/event/billing/cycle/fold` event for auditing purposes.
5. If successful, returns the POIDs of the `/account` object and the `/event/billing/product/fee/cycle/fold` event.

About Applying Folds Only For Account-Level Products

If a fold is configured at account level, the fold is applicable to all the services in the account. The fold is then applied to the first service that is retrieved. If the resource balance is non-zero after applying the fold to the first service, the fold is applied again to the next service until the resource balance is zero.

To apply the fold only to account-level products, you can configure an account-level balance group to track the resource that you want to fold. The fold is then applied to only the products associated with the account-level balance group and not to all the services in the account.

Customizing How Folds Are Applied

To customize how folds are applied, use the `PCM_OP_SUBSCRIPTION_POL_PRE_FOLD` policy opcode.

By default, this policy opcode is an empty hook provided to facilitate any customization before folding the currency or non-currency resources. For example, when billing is run, this policy opcode is called to verify that the `pin_cycle_fees` utility has applied cycle fees to an account.

Customizing the Order to Apply Folds

To customize the order in which folds are applied, use the `PCM_OP_SUBSCRIPTION_POL_SPEC_FOLD` policy opcode.

By default, resources are folded in ascending order based on the resource ID. This policy opcode enables you to change the order in which resources are folded.

For example, you can fold resources in descending order of the resource IDs. To do this, sort the `PIN_FLD_BALANCES` array and return the sorted array.

The following example shows the `PIN_FLIST_SORT` statement for sorting the balances array in descending order:

```
*out_flistp = PIN_FLIST_COPY(i_flistp, ebufp);
s_flistp = PIN_FLIST_CREATE(ebufp);
PIN_FLIST_ELEM_SET(s_flistp, (void *)NULL, PIN_FLD_BALANCES, PIN_ELEMID_ANY,
ebufp);

PIN_FLIST_SORT(*out_flistp, s_flistp, 1, ebufp);
```

This policy opcode returns the contents of the input flist including the POID of the `/balance_group` object and the balance array sorted in ascending order.

Specifying Which Resources to Fold

You can specify which resources are impacted by fold events. This enables you to exclude folding resources that do not need to be folded.

To specify the resources to fold, you set the `PIN_FLD_APPLY_MODE` parameter for the resources in the resource configuration file (`BRM_Home/sys/data/pricing/example/pin_beid`).

The `PIN_FLD_APPLY_MODE` parameter can take the following values:

- `1` indicates the resource will be folded, if appropriate.
- `0` indicates the resource will not be folded.

By default, folds are enabled for all resources in the `pin_beid` file.

After modifying the `pin_beid` file, load the contents of the file into the `/config/beid` object in the BRM database by running the `load_pin_beid` utility.

For more information, see the `pin_beid` file and "[load_pin_beid](#)".

Customizing Which Resources to Fold When Products Are Canceled

The `PCM_OP_SUBSCRIPTION_POL_PREP_FOLD` policy opcode prepares the list of resources that must be folded when a product is canceled. This opcode is called when a product is canceled, and it performs the following functions:

- Checks the canceled product.
- Creates a list of resources affected by the cancellation that must be folded.
- Calls `PCM_OP_SUBSCRIPTION_CYCLE_FOLD` and passes the list of resources as the input.

If you do not want to fold resources after a product is canceled, customize this policy opcode by removing or commenting out the code in the `fm_subscription_pol_prep_fold.c` file.

Assigning Rate Plans and Impact Categories to Events

Use the `PCM_OP_ACT_POL_SPEC_RATES` policy opcode to map an event to a rate plan and impact category.

Note: Ensure that your rate plan structure uses custom event analysis for the event.

The opcode that calls the `PCM_OP_ACT_POL_SPEC_RATES` policy opcode passes in the inherited information for an event. For example, to rate a password, the `PCM_OP_CUST_SET_PASSWD` opcode specifies the inherited password information. The `PCM_OP_ACT_POL_SPEC_RATES` policy opcode specifies that the password is to be rated.

Though most custom rating functionality is now in the rate plan selector, some special cases require you to use the `PCM_OP_ACT_POL_SPEC_RATES` policy opcode. In such a case, you must:

1. Map your custom events to BRM opcodes in the `BRM_Home/sys/data/config/pin_spec_rates` file, and then load the file into the BRM database by using the `load_pin_spec_rates` utility.
2. Define your custom impact categories in the `BRM_Home/sys/data/config/pin_impact_category` file, and then load the file into the BRM database by using the `load_pin_impact_category` utility.

This configures the `PCM_OP_ACT_POL_SPEC_RATES` policy opcode and the rate plan selector to return an event's rate plan and impact category to real-time rating.

Rating an Event Based on Extended Rating Attributes

An event can be rated based on extended rating attributes (ERAs) by using either the usage type or the ERA label, as follows:

- **Usage type:** You can select a specific rate plan and impact category based on the usage type of an event, such as ERAs. For example, if the usage type for an event is Friends and Family, you can rate that event with a special rate of \$1.00 a minute.
- **ERA label:** You can select a specific rate plan based on the ERA label. A label is a separate list of ERA values. You can have multiple labels for a single ERA and use a rate plan selector to choose a different rate based on the label.

For example, you can have a friends and family ERA with two labels, MYFRIENDS and MYFAMILY, and use the rate plan selector to use a different rate for each list.

For more information on ERAs, see ["About Extended Rating Attributes"](#).

To rate an event by using special rates based on an ERA, PCM_OP_ACT_USAGE calls the PCM_OP_RATE_POL_PRE_RATING policy opcode before calling PCM_OP_RATE_EVENT.

The PCM_OP_RATE_POL_PRE_RATING policy opcode calls the PCM_OP_RATE_POL_PROCESS_ERAS policy opcode, which in turn calls the PCM_OP_RATE_GET_ERAS opcode.

PCM_OP_RATE_GET_ERAS retrieves the valid ERAs for an event by performing the following operations:

1. Gets the service and account data for the event from the input flist.
2. Finds the ERA information in the associated `/profile/serv_extrating` or `/profile/acct_extrating` object. The ERA information includes the ERA name, label names, label values, and validity dates.

Note: By default, BRM includes all ERAs in this process, but you can improve real-time rating performance by configuring BRM to omit checking for account-level ERAs, service-level ERAs, or both during rating.

3. Identifies profile sharing groups that the service or subscription service for the event belongs to. If membership is found and the group is active, the opcode retrieves the ERA information.
4. For shared profile groups, finds ERA information in the associated `/group/sharing/profiles` object. The ERA information includes the ERA name, label names, label values, and validity dates.
5. Checks whether the ERA is valid for the event start or end time, based on the configuration.

Note: The configuration for using event start time for the validity check is determined by the `use_prods_based_on_start_t` entry in the CM configuration file. See ["Using Event Start Time to Determine Product's Validity"](#).

6. If the event time is less than the effective time of any of the profiles, searches for that profile in the audit tables to get the correct data matching the event time.

If the ERA is found to be not valid, the opcode skips the remaining steps.

7. If the service has a subscription object, reads the service profiles of that subscription object. If the same ERA is found in the service and subscription service, the service ERA takes precedence.

8. Returns the names of valid ERAs to the PCM_OP_RATE_POL_PROCESS_ERAS policy opcode.

The PCM_OP_RATE_POL_PROCESS_ERAS policy opcode uses the data retrieved by PCM_OP_RATE_GET_ERAS to do the following:

9. Compares certain predefined event fields to the values for the ERAs found in the service or account.

The specific fields compared depend on the service. By default, the policy opcode is configured to work with a GSM telco service and to compare values in the PIN_FLD_CALLING_FROM and PIN_FLD_CALLED_TO fields.

The policy opcode can be easily configured to work with a GPRS service. In that case, it compares values in the PIN_FLD_ANI and PIN_FLD_DN fields.

Note: Using the PCM_OP_RATE_POL_PROCESS_ERAS policy opcode with services other than GSM and GPRS requires customization. See "[Customizing Real-Time Rating for Friends and Family ERAs](#)".

10. Populates the PIN_FLD_USAGE_TYPE field with the names of the valid ERAs.

If multiple ERAs are found for service and account profiles, the ERAs can be combined in the PIN_FLD_USAGE_TYPE field. For example, if Closed User Group (CUG) and Friends & Family (FF) are the qualified ERAs for the event, PIN_FLD_USAGE_TYPE can be set to FF_CUG. By default, PIN_FLD_USAGE_TYPE is populated based on the following qualified ERAs:

- Friends and Family (FF)
- Closed User Group (CUG)
- FF + CUG (FF_CUG)

You can customize the policy opcode to add other ERAs.

11. Populates the PIN_FLD_PROFILE_LABEL_LIST field with ERA label names. Multiple names are separated by a comma, unless you specified a different separator in the opcode and the rate plan selector.

12. Returns the output to the PCM_OP_RATE_POL_PRE_RATING policy opcode.

For specific information about how BRM rates friends and family ERAs, see "[About Rating Based on Friends and Family ERAs](#)".

The PCM_OP_RATE_POL_PRE_RATING policy opcode returns the output list with the PIN_FLD_USAGE_TYPE and PIN_FLD_PROFILE_LABEL_LIST fields to the calling opcode.

The event is then rated or discounted based on the ERA value in either PIN_FLD_USAGE_TYPE or PIN_FLD_PROFILE_LABEL_LIST.

If there are multiple valid ERAs and a rate plan selector is being used for the event, the rating or discounting opcode uses the rate plan selector to determine which ERA should be used for rating or discounting. The rate plan selector would use either `EVENT.PIN_FLD_USAGE_TYPE` or `EVENT.PIN_FLD_PROFILE_LABEL_LIST` to determine a rate based on the ERA.

Modifying Rated Events

Use the `PCM_OP_RATE_POL_POST_RATING` policy opcode to modify rated `/event` objects (for example, change the G/L ID of an event). This policy opcode is called by `PCM_OP_ACT_USAGE`.

The input flist matches the `/event` object that you are modifying. The output flist contains the event field to be changed in the rated object.

The following example shows how to change the G/L ID of an event:

```
STORABLE CLASS /event {
DESCR = "Objects of the event class are created to record the various "
"system-initiated and user-initiated events. The events are "
"either related to a specific service or to an account. The "
"event includes generic information such as start and end "
"times as well the various charges that are incurred by the "
"account due to this event.";
SEQ_START = "1";
INT32 PIN_FLD_GL_ID {
DESCR = "GLID associated with this balance impact. "
"Don't care if 0.";
ORDER = 0;
CREATE = Optional;
MODIFY = Writable;
}
} CREATE = Optional;
MODIFY = Writable;
```

Caution: Before calling the `PCM_OP_RATE_POL_POST_RATING` opcode, the `PCM_OP_ACT_USAGE` opcode stores the balance of a rated event in a balance group cache rather than in the BRM database. If you customize the `PCM_OP_RATE_POL_POST_RATING` opcode to call a standard opcode to change the balance of the rated event directly in the database or in a different cache, the balance group cache from which the `PCM_OP_ACT_USAGE` opcode fetches the final balance impact of the event may not reflect the change made by the `PCM_OP_RATE_POL_POST_RATING` opcode.

Customizing How to Calculate RUMs

Use the `PCM_OP_ACT_POL_SPEC_CANDIDATE_RUMS` policy opcode to define the customized policy to specify the candidate RUM.

A candidate RUM specifies the name, quantity, and unit of measurement for each resource used in the event. For example, an IP fax event's RUM might have the information in [Table 7-2](#):

Table 7-2 Example

Name	Quantity	Unit
duration	2	minutes
page_count	10	pages
byte_count	9568	bytes

By default, the PCM_OP_ACT_POL_SPEC_CANDIDATE_RUMS policy opcode copies PIN_FLD_POID and the PIN_FLD_CANDIDATE_RUMS array to the output flist.

You can change the rules to calculate data by using `/config/rum`. By default, the PCM_OP_ACT_POL_SPEC_CANDIDATE_RUMS policy opcode supports only simple-arithmetic expressions (addition, subtraction, multiplication, and division), but you can add more complex rules to get the event quantity.

If you add a candidate RUM that has the same name as any RUM in the input flist, drop the one from the input flist to ensure that the RUM names are unique.

Rating and Recording Activity Events

BRM uses the PCM_OP_ACT_ACTIVITY opcode to record and rate activity events. This enables BRM to record any type of activity event and any details specific to the event type. Any type of user action can be recorded as an activity event, but it is especially designed to represent events that occur at a single point in time. All activity events are recorded in the system simultaneously.

Note: For events related to customer service usage over a period, the PCM_OP_ACT_START_SESSION and PCM_OP_ACT_LOGIN opcodes are called. For more information on the PCM_OP_ACT_START_SESSION and PCM_OP_ACT_LOGIN opcodes, see *BRM Developer's Reference*.

PCM_OP_ACT_ACTIVITY records events for either `/account` or `/service` objects. The `/account` POID must be specified in both cases.

- When an `/account` POID is used alone, PCM_OP_ACT_ACTIVITY records the event for an `/account` object.
- When both an `/account` POID and a `/service` POID are specified, PCM_OP_ACT_ACTIVITY records the event for a `/service` object.
- When a `NULL` `/service` POID is specified, PCM_OP_ACT_ACTIVITY records an `/account` event.

PCM_OP_ACT_ACTIVITY records event details in the following ways:

- Using the generic `/event/activity` object, the PIN_FLD_DESCR field specifies details of the event in ASCII text format.
- If this is insufficient, an inherited object type can be created by the user from the `/event/activity` object and additional fields added to describe a specific type of event in more detail. When an event of that type occurs, the input flist to this operation must use the PIN_FLD_OBJ_TYPE field to specify the type of event object to create and to include the detailed information fields in the PIN_FLD_INHERITED_INFO substructure. This enables any amount of detail to be recorded for any number of event types.

Note: The `PIN_FLD_INHERITED_INFO` feature requires you to supply a new storable class definition. You determine the fields necessary in the new inherited storable class type and define them by using `PIN_MAKE_FLD`.

BRM controls what `PCM_OP_ACT_ACTIVITY` returns by using the following flags:

- **PCM_OPFLG_READ_RESULT**

When this flag is set, `PCM_OP_ACT_ACTIVITY` returns all fields in the event object, in addition to the POID.

- **PCM_OPFLG_CALC_ONLY**

- When this flag is set, `PCM_OP_ACT_ACTIVITY` only calculates the rated amount and does not record the event in the BRM database. The opcode returns the fields that would have been used to create the event object. No fields in the database are changed, and the event object is not actually created.
- When this flag is *not* set, `PCM_OP_ACT_ACTIVITY` creates the `/event/activity` object to record details of the operation.

Managing Sessions

BRM stores information about sessions in `/event/session` objects in the BRM database.

A session event differs from an activity event in that the start and end of the session event are recorded separately. A session is designed to efficiently track a customer connecting to some type of service for a period of time. For events that occur at a single point in time, the `PCM_OP_ACT_ACTIVITY` opcode is called. For more information on the `PCM_OP_ACT_ACTIVITY` opcode, see *BRM Developer's Reference*.

You use the following Activity standard opcodes to manage sessions:

- `PCM_OP_ACT_LOGIN`. See ["Starting Sessions"](#).
- `PCM_OP_ACT_START_SESSION`. See ["Recording the Start of a Session"](#).
- `PCM_OP_ACT_UPDATE_SESSION`. See ["Updating a Session Event"](#).
- `PCM_OP_ACT_LOGOUT`. See ["Recording the End of a Session"](#).
- `PCM_OP_ACT_END_SESSION`. See ["Rating and Recording Session Events"](#).
- `PCM_OP_ACT_LOAD_SESSION`. See ["Loading Sessions in Real Time"](#).

Starting Sessions

Use the `PCM_OP_ACT_LOGIN` opcode to start a customer session.

`PCM_OP_ACT_LOGIN` performs the following operations:

1. Calls the `PCM_OP_ACT_FIND_VERIFY` opcode to verify the customer's identity.
2. Calls the `PCM_OP_ACT_START_SESSION` opcode to start a login session. See ["Recording the Start of a Session"](#).
3. Returns a flist with the user's `/account` POID.

This opcode is usually called by `PCM_CONNECT`, but it is also called directly. If called by `PCM_CONNECT`, this opcode ignores the database number in the application's `pin.conf` file and searches all schemas for the account.

Recording the Start of a Session

Use the `PCM_OP_ACT_START_SESSION` opcode to record the start of a session event.

This opcode records any type of session event object including details specific to the event type. This opcode does not perform rating. Fees for the session event are calculated and applied when the session ends.

Sessions can be started for either `/account` or `/service` objects. The `/account` object must be specified for both cases.

- When the `/account` object is used alone, `PCM_OP_ACT_START_SESSION` starts a session for the `/account` object.
- When both `/account` and `/service` object POIDs are specified, `PCM_OP_ACT_START_SESSION` starts a session for the `/service` object.
- When a `NULL /service` object is specified, `PCM_OP_ACT_START_SESSION` starts a session for the `/account` object.

Session details can be recorded in the following ways:

- In the generic `/event/session` object, the `PIN_FLD_DESCR` field specifies details of the session in a text format.
- If that is insufficient, you can create an inherited object type from the `/event/session` object and add fields to describe a specific type of session in more detail.

When a session of the specified type occurs, the input flist to `PCM_OP_ACT_START_SESSION` can specify the type of event object to create with the `PIN_OBJ_TYPE` field and include the detailed information fields in the `PIN_FLD_INHERITED_INFO` substructure. This enables any amount of detail to be recorded for any number of session types.

Within the `PIN_FLD_INHERITED_INFO` array, the program supplies a new storable class definition. You determine the fields necessary in the new inherited storable class type and define them by using `PIN_MAKE_FLD`. This procedure is explained in the `pcm.h` file. `PCM_OP_ACT_START_SESSION` then automatically creates storable class definitions and supplies updated release files containing the new storable class definitions.

BRM controls how `PCM_OP_ACT_START_SESSION` records the start of a session event by using the following flags:

- **PCM_OPFLG_READ_RESULT**
When this flag is set, `PCM_OP_ACT_START_SESSION` returns all fields in the event object in addition to the `/account` POID.
- **PCM_OPFLG_CALC_ONLY**
 - When this flag is set, `PCM_OP_ACT_START_SESSION` returns the fields that would have been used to create the event object. No fields in the database are changed, and the event object is not actually created.
 - When the flag is *not* set, `PCM_OP_ACT_START_SESSION` creates an `/event/session` object to record the details of the session event.

Updating a Session Event

Use the `PCM_OP_ACT_UPDATE_SESSION` opcode to update information in a session event.

Note: The session event object must already have been created by a call to PCM_OP_ACT_START_SESSION or by an opcode that calls that opcode.

This opcode updates the PIN_FLD_DESCR field and any inherited data fields in the session object. Only the fields specified in the opcode's input list are updated; all other fields in the **/event/session** object are left unchanged. This opcode uses the generalized PIN_FLD_INHERITED_INFO substruct so it can update **/event/session** objects of any type.

Recording the End of a Session

Use the PCM_OP_ACT_LOGOUT opcode to record the end of a login session. This opcode calls the PCM_OP_ACT_END_SESSION opcode to end the session and assess any changes.

Rating and Recording Session Events

Use the PCM_OP_ACT_END_SESSION opcode to rate and record the end of a session event.

PCM_OP_ACT_END_SESSION performs the following operations:

1. Records the session's ending time stamp.
2. Updates the PIN_FLD_DESCR field and any inherited data fields in the **/event/session** object.
3. Returns the following, depending on the flags passed in:
 - When the PCM_OPFLG_READ_RESULT flag is set, the opcode returns all fields in the event object, in addition to the POID.
 - When the PCM_OPFLG_CALC_ONLY flag is set, the opcode returns the fields that would have been used to create the event object. No fields in the database are changed, and the event object is not actually created.
 - When no flags are set, the opcode returns the POID of the event object.

Loading Sessions in Real Time

Use the PCM_OP_ACT_LOAD_SESSION opcode to create, rate, and record a session event as a single operation. This opcode is valuable as a tool to load sessions in real time.

Managing Price Lists

You specify how to charge for the services and products offered by your company by creating a price list. This data is then stored in the BRM database as pricing objects, such as **/deal**, **/plan**, and **/product** objects.

For information on how BRM uses price lists, see ["About Creating a Price List"](#).

You can create, retrieve, modify, or delete price list data individually or globally.

- To create, modify, or delete all objects in a price list, use the PCM_OP_PRICE_SET_PRICE_LIST opcode. See ["Committing Price List Data to the BRM Database"](#).

- To retrieve all objects in a price list, use the PCM_OP_PRICE_GET_PRICE_LIST opcode. See ["Retrieving Price List Data from the BRM Database"](#).
- To create, modify, or delete individual pricing objects, use the PCM_OP_PRICE_COMMIT_* standard opcodes. See ["Managing Individual Pricing Objects"](#).

Committing Price List Data to the BRM Database

Use the PCM_OP_PRICE_SET_PRICE_LIST opcode to commit all data for a price list, including plans, products, offer profiles, and deals, to the BRM database. This opcode is called directly by Pricing Center, Customer Center, your custom client application, or the **loadpricelist** utility to load price list data into the database.

PCM_OP_PRICE_SET_PRICE_LIST creates one input flist that contains information for creating, modifying, or deleting the following pricing objects:

- **/best_pricing**
- **/deal**
- **/dependency**
- **/discount**
- **/offer_profile**
- **/plan**
- **/product**
- **/sponsorship**
- **/transition**

PCM_OP_PRICE_SET_PRICE_LIST performs the following in one transaction:

- Verifies the relationships and dependencies between the pricing objects
- Commits all objects to the BRM database
- Prepares a list of all **/sponsorship** objects that were created or modified and generates the **/event/notification/price/sponsorships/modify** notification event
- Prepares a list of all **/product** objects that were created or modified and generates the **/event/notification/price/products/modify** notification event
- Prepares a list of all **/offer_profile** objects that were created or modified and generates the **/event/notification/price/offer_profile/modify** notification event
- Prepares a list of all **/discount** objects that were created or modified and generates the **/event/notification/price/discounts/modify** notification event

Retrieving Price List Data from the BRM Database

Use the PCM_OP_PRICE_GET_PRICE_LIST opcode to retrieve price list data from the BRM database. This opcode is called directly by the following applications to retrieve price list data:

- Pricing Center
- Customer Center
- A custom client application
- **loadpricelist**, when used with the **-r** parameter

PCM_OP_PRICE_GET_PRICE_LIST retrieves data about the following pricing objects, rebuilds the product flist, and then returns the data back to the calling application:

- **/best_pricing**
- **/deal**
- **/dependency**
- **/discount**
- **/offer_profile**
- **/plan**
- **/product**
- **/sponsorship**
- **/transition**

By default, this opcode returns all price list data. However, you can specify that the opcode retrieve only **/product**, **/discount**, or **/sponsorship** objects by passing optional input flist fields:

- To retrieve only **/product**, **/discount**, or **/sponsorship** objects, pass the PIN_FLD_FLAGS input flist field set to **0x10** for **/product** objects, **0x20** for **/discount** objects, and **0x40** for **/sponsorship** objects. You can specify multiple object types by adding these values; for example, set PIN_FLD_FLAGS to **0x30** to retrieve both **/product** and **/discount** objects. You can also set the field to **0x0008** to retrieve tailor-made products or other customized pricing data from the BRM system.
- To retrieve **/product** and **/discount** objects based on the service type, pass the PIN_FLD_PERMITTED input flist field set to the desired service type, such as **/service/email**.
- To retrieve **/product**, **/discount**, or **/sponsorship** objects based on the object modification time, pass the PIN_FLD_MOD_T input flist field set to the desired time stamp.

If the opcode retrieves customized pricing data, the output flist includes the PIN_FLD_TAILORMADE field in the product array with a value of **1**. The output flist also includes PIN_FLD_TAILORMADE_DATA, which lists the resources modified by the customization and the percentage changes.

The purchase, cycle, and usage validity periods of products and discounts are returned in the PIN_FLD_PRODUCTS and PIN_FLD_DISCOUNTS arrays in the output flist. If the purchase, cycle, or usage period has a relative start or end time, the relative offset information is returned in respective START_UNIT and START_OFFSET or END_UNIT and END_OFFSET fields. The unit fields specify the type of offset unit, such as minutes, days, or accounting cycles. The offset fields specify the number of offset units in the relative period.

If the purchase, cycle, or usage period starts on first usage (when the customer first uses the product or discount) and the validity period has not yet been set, the START_OFFSET field has a value of **-1**.

The validity period of resources is returned in the PIN_FLD_BAL_IMPACTS array in the PIN_FLD_RATES array. If the resource has a relative start or end time, the relative offset information is returned in the PIN_FLD_RELATIVE_START_UNIT and PIN_FLD_RELATIVE_START_OFFSET or PIN_FLD_RELATIVE_END_UNIT and PIN_FLD_RELATIVE_END_OFFSET fields. For resources, the unit fields specify only cycles. The offset fields specify the number of offset cycles in the relative period.

If the resource starts on first usage (when the customer consumes the resource for the first time) and the validity period has not yet been set, the `PIN_FLD_RELATIVE_START_OFFSET` field has a value of `-1`

Managing Individual Pricing Objects

You can manage an individual pricing object, such as a `/deal` object, by using the `PCM_OP_PRICE_COMMIT_*` standard opcodes. These opcodes create, modify, or delete objects by using data in the input flist only; they do not verify relationships or dependencies with other pricing objects before committing changes to the database. For example, the `PCM_OP_PRICE_COMMIT_PRODUCT` opcode creates a `/product` object without first verifying that it is associated with a deal in the price list.

Important: Because these opcodes do not verify dependencies with other pricing objects, you should use them in development environments only. To create pricing objects in a production environment, use the `PCM_OP_PRICE_SET_PRICE_LIST` opcode. See ["Committing Price List Data to the BRM Database"](#).

You use the following opcodes to manage individual pricing objects:

- `PCM_OP_PRICE_COMMIT_PRODUCT`. See ["Managing /product Objects"](#).
- `PCM_OP_PRICE_COMMIT_DISCOUNT`. See ["Managing /discount Objects"](#).
- `PCM_OP_PRICE_COMMIT_DEAL`. See ["Managing /deal Objects"](#).
- `PCM_OP_PRICE_COMMIT_PLAN`. See ["Managing /plan Objects"](#).
- `PCM_OP_PRICE_COMMIT_DEPENDENCY`. See ["Managing /dependency Objects"](#).
- `PCM_OP_PRICE_COMMIT_TRANSITION`. See ["Managing /transition Objects"](#).
- `PCM_OP_PRICE_COMMIT_PLAN_LIST`. See ["Managing /group/plan_list Objects"](#).
- `PCM_OP_PRICE_COMMIT_SPONSORSHIP`. See ["Managing /sponsorship Objects"](#).

Note: Each of these standard opcodes calls `PREP` and `VALID` policy opcodes before committing changes to the database. You can use these policy opcodes to add data to the objects or to perform custom validation.

Managing /product Objects

Use the `PCM_OP_PRICE_COMMIT_PRODUCT` opcode to create, modify, or delete `/product` objects in the BRM database.

To create or modify `/product` objects, pass details about the product in the opcode's `PIN_FLD_PRODUCT` input flist array. In the array, you must also pass in the `PIN_FLD_CODE` field set to the `/product` object's unique identifier and the `PIN_FLD_NAME` field set to the product's name, which is modifiable.

This opcode validates and commits the following objects from a product flist: `/rate`, `/rate_plan`, `/rate_plan_selector`, and `/rollover`. Products are created or modified and, if the delete flag is sent in, deleted.

/rate_plan objects can have multiple rates, one for each rate tier (priority), date, day, and time-of-day range. These ranges are structured into a four-level tree of rate tier, date, day, and time. An optional rate plan selector is used when multiple rate plans from which to select are available.

When a product grants a non-currency resource, such as free minutes, the validity period of the granted resource is stored in the **/rate** object's `PIN_FLD_BAL_IMPACTS` array. See ["Managing the Validity Period of Granted Resources"](#).

`PCM_OP_PRICE_COMMIT_PRODUCT` performs the following in one transaction:

- Generates the **/event/notification/price/products/modify** notification event
- Commits all objects to the BRM database
- Returns the following in the output list:
 - If all operations are successful, this opcode returns `PIN_FLD_RESULTS` set to **1** and `PIN_FLD_PRODUCTS` set to the **/product** POID.
 - If any operation fails, this opcode returns `PIN_FLD_RESULTS` set to **0**. It also returns two additional fields: `PIN_FLD_RESULT_FORMAT`, which lists all error codes, and `PIN_FLD_DESCR`, which provides a concise error description.

Managing the Validity Period of Granted Resources

Resource validity periods define when a granted resource is available for consumption.

To set a resource's start and end times, pass the following values in the `PIN_FLD_BAL_IMPACTS` array in the `PIN_FLD_RATES` array of the `PCM_OP_PRICE_COMMIT_PRODUCT` input list:

- Set the `PIN_FLD_FLAGS` field to `PIN_RATE_BAL_FLG_GRANTABLE (0x08)`. This specifies that the balance impact is granting the resource.
- Specify the start times as follows:
 - To start on a specific date, pass the date in `PIN_FLD_START_T`.
 - To start immediately, set the value of `PIN_FLD_START_T` to **0**.
 - To start on first usage (when the customer impacts the resource balance for the first time), set the value of `PIN_FLD_RELATIVE_START_OFFSET` to **-1**.
 - To start relative to the grant date, set the value of `PIN_FLD_RELATIVE_START_UNIT` to the type of relative unit and set the number of units in `PIN_FLD_RELATIVE_START_OFFSET`.
- Specify the end time as follows:
 - To never end, set the value of `PIN_FLD_END_T` to **0**.
 - To end relative to the start date, set the value of `PIN_FLD_RELATIVE_END_UNIT` to the type of relative unit and set the number of units in `PIN_FLD_RELATIVE_END_OFFSET`.

For more information about how to specify the unit and offset values, see ["Specifying Relative Start and End Times for Granted Resources"](#).

When the deal is committed to the BRM database, any relative validity information is stored in the `PIN_FLD_START_DETAILS` and `PIN_FLD_END_DETAILS` fields in the **/rate** object's `PIN_FLD_BAL_IMPACTS` array.

When the resource is granted to an account, any relative validity information is stored in the `PIN_FLD_VALID_FROM_DETAILS` and `PIN_FLD_VALID_TO_DETAILS` fields in the account `/balance_group` object's `PIN_FLD_BAL_IMPACTS` array.

For information about the values stored in details fields, see ["Storing Start and End Times for Granted Resources"](#).

Specifying Relative Start and End Times for Granted Resources

Relative start and end date information for granted resources is passed in `UNIT` and `OFFSET` fields:

- `PIN_FLD_RELATIVE_START_UNIT`
- `PIN_FLD_RELATIVE_START_OFFSET`
- `PIN_FLD_RELATIVE_END_UNIT`
- `PIN_FLD_RELATIVE_END_OFFSET`

The `UNIT` fields specify the type of offset unit, and the offset fields specify the number of units in the relative period. The `UNIT` fields can be one of the following values:

- 1 = Seconds
- 2 = Minutes
- 3 = Hours
- 4 = Days
- 5 = Months
- 8 = Accounting cycles
- 9 = Event cycles

The `UNIT` and `OFFSET` fields are used in combination to determine the relative offset period. For example:

- If `PIN_FLD_RELATIVE_START_UNIT` has a value of **8** (accounting cycles) and `PIN_FLD_RELATIVE_START_OFFSET` has a value of **2**, the granted resource becomes valid (available for consumption) two accounting cycles after the resource is granted.
- If `PIN_FLD_RELATIVE_END_UNIT` has a value of **5** (months) and `PIN_FLD_RELATIVE_END_OFFSET` has a value of **2**, the validity period ends two months from the date it becomes valid.

You can specify any number of units in the `OFFSET` fields, up to 1048576. This is equivalent to approximately 12 days in seconds, 728 days in minutes, and 119 years in hours.

Storing Start and End Times for Granted Resources

The validity period information for granted resources that is passed in the unit and offset fields of opcode flists is stored in the `/rate` object's `PIN_FLD_BAL_IMPACTS` array in these details fields:

- `PIN_FLD_START_DETAILS`
- `PIN_FLD_END_DETAILS`

The details fields are 32-bit integer fields that store the following values:

- **Mode:** The mode specifies generally when the validity period starts or ends. The mode also helps to determine how the start and end times are set in the account's

/balance_group object when the resource is granted. The mode is stored in the lower 8th bit and can have one of the following values:

For START-DETAILS:

- **0 = PIN_VALIDITY_ABSOLUTE:** The validity period starts on the date specified in PIN_FLD_START_T. When the resource is granted, the start date is set in the **/balance_group** object's PIN_FLD_VALID_FROM field.
- **1 = PIN_VALIDITY_IMMEDIATE:** The validity period starts when the resource is granted. The **/balance_group** object's PIN_FLD_VALID_FROM field is set to the grant date.
- **3 = PIN_VALIDITY_FIRST_USAGE:** The validity period starts when the resource balance is first impacted by an event.
- **4 = PIN_VALIDITY_RELATIVE:** The validity period starts relative to the grant date. When the resource is granted, the start date is calculated by adding the relative offset period to the grant date. The start date is then set in the **/balance_group** object's PIN_FLD_VALID_FROM field.

For END-DETAILS:

- **0 = PIN_VALIDITY_ABSOLUTE:** The validity period ends on the date specified in PIN_FLD_END_T. When the resource is granted, the end date is set in the **/balance_group** object's PIN_FLD_VALID_TO field.
- **2 = PIN_VALIDITY_NEVER:** The validity period never ends. When the resource is granted, the **/balance_group** object's PIN_FLD_*_END_T field is set to 0.
- **4 = PIN_VALIDITY_RELATIVE:** The validity period ends relative to when it starts. When the resource is granted, the end date is calculated by adding the relative offset period to the start date. The end date is then set in the **/balance_group** object's PIN_FLD_VALID_TO field.

Note: When the resource starts on first usage, the mode of the end time can be only PIN_VALIDITY_NEVER or PIN_VALIDITY_RELATIVE.

When the resource is granted and its validity is activated, PIN_FLD_VALID_FROM and PIN_FLD_VALID_TO are set in the **/balance_group** object, and the mode in the PIN_FLD_VALID_FROM_DETAILS and PIN_FLD_VALID_TO_DETAILS fields in the **/balance_group** object are set to PIN_VALIDITY_ABSOLUTE.

- **Relative offset unit:** This value is set when the resource balance starts at a time relative to when it is granted or when the resource balance ends at a time relative to the start time. It specifies the type of offset unit and corresponds to the value of PIN_FLD_RELATIVE_START_UNIT or PIN_FLD_RELATIVE_END_UNIT that is passed in opcode flists. The relative offset unit is stored in the next four bits of the details field.
- **Number of offset units:** This value specifies how many offset units are in the relative period and corresponds to the value of PIN_FLD_RELATIVE_START_OFFSET or PIN_FLD_RELATIVE_END_OFFSET that is passed in opcode flists. The number of offset units is stored in the remaining 20 bits of the details field.

Customizing How to Create and Delete Products

To customize how to create and delete products:

- To enable data modification during **/product** object creation, use the PCM_OP_PRICE_POL_PREP_PRODUCT policy opcode. Use this policy opcode to enhance **/product** objects with additional data not provided by either the GUI application or by the Price List FM.

The PCM_OP_PRICE_COMMIT_PRODUCT opcode calls this policy opcode before creating a new **/product** object.

Note: This policy opcode is called before the Price List FM performs any validation checks.

- To enable validation during **/product** object creation, use the PCM_OP_PRICE_POL_VALID_PRODUCT policy opcode. This policy opcode can be used to perform validations in addition to those performed by the Price List FM.

The PCM_OP_PRICE_COMMIT_PRODUCT opcode calls this policy opcode before creating a **/product** object. By default, this policy opcode is an empty hook that returns the result PIN_PRICE_VERIFY_PASSED.

- To verify that deleting a **/product** object is permitted, use the PCM_OP_PRICE_POL_DELETE_PRODUCT policy opcode. Use this policy opcode to perform validations in addition to those performed by the Price List FM.

PCM_OP_PRICE_COMMIT_PRODUCT calls this policy opcode before deleting the **/product** object. By default, it returns the result PIN_PRICE_VERIFY_PASSED.

Success or failure is indicated by the value returned in the PIN_FLD_RESULTS output flist field: a value of **1** indicates that the operation succeeded and a value of **0** indicates that the operation failed.

Using Opcodes to Customize Products

You can customize products by changing individual rates and price models for specific resources. Customer service representatives (CSRs) can use this capability in Customer Center, but you can also call opcodes from third-party CRM applications.

Note: Customer Center enforces permissioning requirements for product customization. Permissioning is not enforced when you customize products by calling opcodes directly. You must implement permissioning in the CRM tool that calls the opcodes.

Creating a Customized Product

To create custom products:

- Call the PCM_OP_PRICE_GET_PRODUCT_INFO opcode to retrieve information about the base product, based on its POID.
- Based on information about the base product and on the customization information (resources modified and the percentage modification), use the PCM_OP_PRICE_PREP_TAILORMADE_PRODUCT opcode to prepare an flist for the new customized **/product** object.
- Call the PCM_OP_SUBSCRIPTION_PURCHASE_DEAL opcode to purchase the customized product and create a **/purchased_product** object to represent it in the

account. This opcode calls the PCM_OP_SUBSCRIPTION_PURCHASE_PRODUCT opcode.

- Pass the flist prepared by PCM_OP_PRICE_PREP_TAILORMADE_PRODUCT to the PCM_OP_PRICE_SET_PRICE_LIST opcode to commit the customized **/product** object to the BRM database. See "[Committing Price List Data to the BRM Database](#)".
- Call the PCM_OP_SUBSCRIPTION_CYCLE_FORWARD opcode to apply the appropriate cycle fees. See "[About Applying Cycle Forward Fees](#)".

Modifying a Customized Product

From Customer Center, the only change you can make to an existing customized product is to modify its validity dates. This limitation does not apply when modifying customized products by calling opcodes directly, however.

- Specify changes to the customized product's rates and price models and pass them to PCM_OP_PRICE_PREP_TAILORMADE_PRODUCT to prepare an flist for the modified customized **/product** object.
- Pass the flist to PCM_OP_PRICE_SET_PRICE_LIST to update the **/product** object. See "[Committing Price List Data to the BRM Database](#)".
- Call the PCM_OP_SUBSCRIPTION_SET_PRODINFO opcode to update the **/purchased_product** object associated with the customized product.

Managing /discount Objects

Use the PCM_OP_PRICE_COMMIT_DISCOUNT opcode to:

- Create, modify, or delete **/discount** objects in the BRM database
- Create pipeline discount models

Note: The opcode cannot modify pipeline discount models.

Creating /discount Objects

To create **/discount** objects, pass details about the discount in the opcode's PIN_FLD_DISCOUNT input flist array. You can also optionally create a pipeline discount model by using the PIN_FLD_PIPELINE_DISC_MODELS input flist array. See "[Creating Pipeline Discount Models](#)".

An offer profile and the provisioning tag for the associated discount use the same name and that name must be unique. If you create an offer profile to associate with existing discount, use the provisioning tag in the discount object to name the offer profile. When you configure a new discount around an existing offer profiles, use the appropriate offer profile name as the provisioning tag for the discount. Use the resource specified in the offer profile as a tracking resource in the discount that is attached to the offer profile.

When the **/discount** object does not exist in the database, PCM_OP_PRICE_COMMIT_DISCOUNT creates a **/discount** object by doing the following:

- Calling the PCM_OP_PRICE_POL_PREP_DISCOUNT policy opcode to perform any custom preparation. By default, this policy opcode is an empty hook, but you can customize it to enhance **/discount** objects with additional data or to perform other custom actions. See "[Customizing /discount Objects](#)".

- Calling the PCM_OP_PRICE_POL_VALID_DISCOUNT policy opcode to perform any custom validation. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions. See "[Customizing /discount Objects](#)".
- Creating the specified **/discount** object in the BRM database.
- Generating the **/event/notification/price/discounts/modify** notification event. This enables you to synchronize discounts between BRM and external CRM applications.

PCM_OP_PRICE_COMMIT_DISCOUNT performs all operations within a single transaction, so success or failure is indicated by the value returned for PIN_FLD_RESULTS:

- When successful, PCM_OP_PRICE_COMMIT_DISCOUNT sets PIN_FLD_RESULTS to **1**.
- When any operation fails, PCM_OP_PRICE_COMMIT_DISCOUNT sets PIN_FLD_RESULTS to **0**. It also returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

Creating Pipeline Discount Models

To create pipeline discount models, pass details about the discount model in the PCM_OP_PRICE_COMMIT_DISCOUNT opcode's PIN_FLD_PIPELINE_DISC_MODELS input flist array. The array can contain the following information:

- Discount model version and configuration
- Discount/chargeshare trigger
- Discount/chargeshare condition
- Discount/chargeshare rules
- Discount/chargeshare master
- Discount/chargeshare detail
- Discount/chargeshare step

The discount model, trigger, rule, step, and master are uniquely identified by a code string in the PIN_FLD_CODE_STR field. If a code string is not passed in the input flist, the opcode generates one automatically.

Modifying /discount Objects

To modify **/discount** objects, pass details about the discount in the opcode's PIN_FLD_DISCOUNT input flist array. In the array, you must also pass in the PIN_FLD_CODE field set to the **/discount** object's unique identifier and the PIN_FLD_NAME field set to the discount's name, which is modifiable.

Important: This opcode overwrites data in existing **/discount** objects, so be sure you pass in the correct object to modify.

When the object already exists in the database, PCM_OP_PRICE_COMMIT_DISCOUNT modifies the **/discount** object by doing the following:

- Overwriting the specified **/discount** object in the BRM database.

- Generating the **/event/notification/price/discounts/modify** notification event. This enables you to synchronize discounts between BRM and external CRM applications.

PCM_OP_PRICE_COMMIT_DISCOUNT performs all operations within a single transaction, so success or failure is indicated by the value returned for PIN_FLD_RESULTS:

- When successful, PCM_OP_PRICE_COMMIT_DISCOUNT sets PIN_FLD_RESULTS to **1**.
- When any operation fails, PCM_OP_PRICE_COMMIT_DISCOUNT sets PIN_FLD_RESULTS to **0**. It also returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

Deleting /discount Objects

To delete **/discount** objects, pass details about the discount in the opcode's PIN_FLD_DISCOUNT input flist array and set the PIN_FLD_DELETED_FLAG input flist field to **1**.

When PIN_FLD_DELETED_FLAG is set, PCM_OP_PRICE_COMMIT_DISCOUNT deletes the **/discount** object by doing the following:

- Calling the PCM_OP_PRICE_POL_DELETE_DISCOUNT policy opcode to perform any custom validation. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.
- Deleting the specified **/discount** object in the BRM database.

PCM_OP_PRICE_COMMIT_DISCOUNT performs all operations within a single transaction, so success or failure is indicated by the value returned for PIN_FLD_RESULTS:

- When successful, PCM_OP_PRICE_COMMIT_DISCOUNT sets PIN_FLD_RESULTS to **1**.
- When any operation fails, PCM_OP_PRICE_COMMIT_DISCOUNT sets PIN_FLD_RESULTS to **0**. It also returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

Customizing /discount Objects

You can customize the **/discount** object before it is committed to the database by using the Price List FM policy opcodes:

- To enable data modification during **/discount** object creation, use the PCM_OP_PRICE_POL_PREP_DISCOUNT policy opcode. Use this policy opcode to enhance **/discount** objects with additional data not provided by either the GUI application or by the Price List FM.

This policy opcode is called by PCM_OP_PRICE_COMMIT_DISCOUNT. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.

Success or failure is indicated by the value returned in the PIN_FLD_RESULTS output flist field: a value of **1** indicates that all operations succeeded and a value of **0** indicates that the operation failed.

- To enable validation during **/discount** object creation, use the PCM_OP_PRICE_POL_VALID_DISCOUNT policy opcode. This policy opcode can be used to enhance **/discount** objects with additional data not provided by either Pricing Center or by other opcodes in the Price List FM.

This policy opcode is called by PCM_OP_PRICE_COMMIT_DISCOUNT before creating a **/discount** object. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.

This policy opcode is called before the Price List FM performs any default validation checks.

Success or failure is indicated by the value returned in the PIN_FLD_RESULTS output flist field: a value of **1** indicates that all operations succeeded and a value of **0** indicates that the operation failed.

- To verify that deleting a **/discount** object is permitted, use the PCM_OP_PRICE_POL_DELETE_DISCOUNT policy opcode. Use this policy opcode to perform validations in addition to those performed by the Price List FM.

This policy opcode is called by PCM_OP_PRICE_COMMIT_DISCOUNT before it deletes a **/discount** object. By default, this policy opcode is an empty hook, but you can customize it to enhance **/discount** objects with additional data or to perform other customizations.

This policy opcode is intended as a place for you to add validation checks or other functionality before allowing a **/discount** object to be deleted. For example, you may want to confirm that the valid time period for the **/discount** object has expired before allowing it to be deleted.

Success or failure is indicated by the value returned in the PIN_FLD_RESULTS output flist field: a value of **1** indicates that all operations succeeded and a value of **0** indicates that the operation failed.

Retrieving Discount Data

Use the PCM_OP_PRICE_GET_DISCOUNT_INFO opcode to retrieve details about the real-time **/discount** object along with the pipeline discount model information from the BRM database. The discount model data includes the following:

- Discount model version and configuration
- Discount/chargeshare trigger
- Discount/chargeshare condition
- Discount/chargeshare rules
- Discount/chargeshare master
- Discount/chargeshare detail
- Discount/chargeshare step
- Balance impact

This opcode takes as input the discount object POID and optionally the discount object name. This opcode performs the following:

1. Retrieves the real-time **/discount** object details from the BRM database.
2. Retrieves the related pipeline discount model information used by the **/discount** object.

3. Returns details about the real-time **/discount** object and pipeline discount model in the output flist.

Managing /deal Objects

Use the PCM_OP_PRICE_COMMIT_DEAL opcode to create, change, or delete **/deal** objects in the BRM database.

This opcode accepts an flist consisting of a PIN_FLD_DEAL array, each element of which represents a **/deal** object.

PCM_OP_PRICE_COMMIT_DEAL determines whether to create, modify, or delete the specified **/deal** object:

- When the object does not exist, this opcode *creates* the specified **/deal** object by doing the following:
 - Calling the PCM_OP_PRICE_POL_PREP_DEAL policy opcode to perform any custom preparation. See "[Customizing How to Create and Delete Deals](#)".
 - Calling the PCM_OP_PRICE_POL_VALID_DEAL policy opcode to perform any custom validation. See "[Customizing How to Create and Delete Deals](#)".
 - Creating the specified object in the BRM database.
- When the object already exists, this opcode *modifies* the specified **/deal** object in the BRM database.

Important: PCM_OP_PRICE_COMMIT_DEAL overwrites data in existing **/deal** objects, so be sure you pass in the correct object to modify.

- When the PIN_FLD_DELETED_FLAG is set, this opcode *deletes* the specified **/deal** object by doing the following:
 - Calling the PCM_OP_PRICE_POL_DELETE_DISCOUNT policy opcode to perform any custom validation. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.
 - Deleting the specified object in the BRM database.

PCM_OP_PRICE_COMMIT_DEAL performs all operations within a single transaction, so success or failure is indicated by the value returned in the PIN_FLD_RESULTS output flist field: a value of **1** indicates that all operations succeeded and a value of **0** indicates that the operation failed. When the operation fails, the opcode also returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

To set the purchase, cycle, and usage start and end dates of the products and discounts in the deal, pass the following values in opcode input flists in the PIN_FLD_PRODUCTS and PIN_FLD_DISCOUNTS arrays. In the following fields, the asterisk (*) indicates either PURCHASE, CYCLE, or USAGE.

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

- To start immediately, set the value of PIN_FLD_*_START_OFFSET to **0**.

- To start on first usage (when the customer uses the product or discount for the first time), set the value of PIN_FLD_*_START_OFFSET to **-1**.
- To start relative to the purchase date, set the relative unit (for example, days or cycles) in PIN_FLD_*_START_UNIT and set the number of relative units in PIN_FLD_*_START_OFFSET.

Note: If you specify a relative unit for more than one period (purchase, cycle, or usage) and you select a different relative unit for each, if one of the relative units is cycles, the PCM_OP_PRICE_POL_VALID_DEAL policy opcode cannot validate that the cycle or usage period falls within the purchase period. The validation for this kind of configuration is performed when the product is purchased by a customer.

- To never end, set the value of PIN_FLD_*_END_OFFSET to **0**.
- To end relative to the start date, set the relative unit (for example, days or cycles) in PIN_FLD_*_END_UNIT and set the number of relative units in PIN_FLD_*_END_OFFSET.

For more information about how to specify the unit and offset values, see ["Specifying Relative Start and End Times for Products and Discounts in a Deal"](#).

If the deal's products or discounts grant resources that start on first usage (when the customer impacts the resource balance for the first time), you can specify that the validity period of all resources in the deal that start on first usage are set when one of those resources is impacted for the first time. To do this, set the PIN_FLD_GRANT_RESOURCES_AS_GROUP flag value in the PIN_FLD_FLAGS field.

When the deal is committed to the BRM database, the purchase, cycle, and usage validity information is stored in the PIN_FLD_*_START_DETAILS and PIN_FLD_*_END_DETAILS fields in the /deal object's product and discount arrays. For information about the values stored in the details fields, see ["Storing Start and End Times for Products and Discounts in a Deal"](#).

Specifying Relative Start and End Times for Products and Discounts in a Deal

When the purchase, cycle, or usage period of a product or discount has a relative start or end time, the details about the relative period are passed in opcode flists in UNIT and OFFSET fields.

The following UNIT fields specify the type of relative unit:

- PIN_FLD_PURCHASE_START_UNIT
- PIN_FLD_PURCHASE_END_UNIT
- PIN_FLD_CYCLE_START_UNIT
- PIN_FLD_CYCLE_END_UNIT
- PIN_FLD_USAGE_START_UNIT
- PIN_FLD_USAGE_END_UNIT

The unit can be one of these values:

- Seconds = 1
- Minutes = 2

- Hours = 3
- Days = 4
- Months = 5
- Accounting cycles = 8

The following OFFSET fields specify the number of units in the relative period:

- PIN_FLD_PURCHASE_START_OFFSET
- PIN_FLD_PURCHASE_END_OFFSET
- PIN_FLD_CYCLE_START_OFFSET
- PIN_FLD_CYCLE_END_OFFSET
- PIN_FLD_USAGE_START_OFFSET
- PIN_FLD_USAGE_END_OFFSET

The UNIT and OFFSET fields for each purchase, cycle, and usage period are used in combination to determine the relative offset period. For example:

- If a product's PIN_FLD_CYCLE_START_UNIT has a value of 8 (accounting cycles) and PIN_FLD_CYCLE_START_OFFSET has a value of 3, the cycle fee starts three accounting cycles after the product is purchased.
- If a discount's PIN_FLD_PURCHASE_END_UNIT has a value of 8 (accounting cycles) and PIN_FLD_PURCHASE_END_OFFSET has a value of 3, the discount expires three accounting cycles after it is activated.

You can specify any number of units in the OFFSET fields, up to 1048576. This is equivalent to approximately 12 days in seconds, 728 days in minutes, and 119 years in hours.

Information about the relative offset validity periods is stored in the BRM database in DETAILS fields in the /deal object's products and discounts arrays. For more information, see ["Storing Start and End Times for Products and Discounts in a Deal"](#).

The product and discount start and end dates (the PIN_FLD_*_START_T and PIN_FLD_*_END_T fields) are not set in the /deal object but are set in the /purchased_product and /purchased_discount objects when the product or discount is purchased.

Storing Start and End Times for Products and Discounts in a Deal

When a product or discount has a relative start or end time, the relative information passed in the PIN_FLD_*_UNIT and PIN_FLD_*_OFFSET fields of opcode flists is stored in the database in PIN_FLD_*_DETAILS fields in the /deal object:

- PIN_FLD_PURCHASE_START_DETAILS
- PIN_FLD_PURCHASE_END_DETAILS
- PIN_FLD_CYCLE_START_DETAILS
- PIN_FLD_CYCLE_END_DETAILS
- PIN_FLD_USAGE_START_DETAILS
- PIN_FLD_USAGE_END_DETAILS

The START-DETAILS and END-DETAILS fields are 32-bit integer fields that store three values: the mode of the validity period, the relative offset unit, and the number of offset units in the relative period:

- **Mode:** The mode specifies generally when the validity period starts or ends. The mode also helps to determine how the start and end dates are set in the account's **/purchased_product** and **/purchased_discount** objects when the product or discount is purchased. The mode is stored in the lower 8th bit and can have one of the following values:

For the START-DETAILS:

- **1 = PIN_VALIDITY_IMMEDIATE:** The validity period starts when the product or discount is purchased. When purchased, the PIN_FLD_*_START_T field in the account's **/purchased_product** or **/purchased_discount** object is set to the purchase time.
- **3 = PIN_VALIDITY_FIRST_USAGE:** The validity period starts when the product or discount is first used to rate the customer's usage.
- **4 = PIN_VALIDITY_RELATIVE:** The validity period starts relative to the purchase date. When the product or discount is purchased, the start date is calculated by adding the relative offset period to the purchase date. The start date is then set in the purchased product's or purchased discount's PIN_FLD_*_START_T field.

For the END-DETAILS:

- **2 = PIN_VALIDITY_NEVER:** The validity period never ends. When the product or discount is purchased, the purchased product's or purchased discount's PIN_FLD_*_END_T field is set to 0.
- **4 = PIN_VALIDITY_RELATIVE:** The validity period ends relative to when it starts. When the product or discount is purchased, the end date is calculated by adding the relative offset period to the start date. The end date is then set in the purchased product's or purchased discount's PIN_FLD_*_END_T field.

When PIN_FLD_*_START_T and PIN_FLD_*_END_T are set in the account's **/purchased_product** and **/purchased_discount** objects, the mode in the details fields of those objects is set to PIN_VALIDITY_ABSOLUTE.

- **Relative offset unit:** This value is set when the product or discount starts at a time relative to the purchase date or ends at a time relative to the start date. It specifies the type of offset unit and corresponds to the value of PIN_FLD_*_START_UNIT or PIN_FLD_*_END_UNIT that is passed in opcode flists. The relative offset unit is stored in the next four bits of the details field and can have one of the following values:

- Seconds = 1
- Minutes = 2
- Hours = 3
- Days = 4
- Months = 5
- Accounting cycles = 8

- **Number of offset units:** This value specifies how many offset units are in the relative period and corresponds to the value of PIN_FLD_*_START_OFFSET or PIN_FLD_*_END_OFFSET that is passed in opcode flists. The number of offset units is stored in the remaining 20 bits of the details field.

Customizing How to Create and Delete Deals

To customize how to create and delete deals, use the following policy opcodes:

- To enable data modification during **/deal** object creation, use the PCM_OP_PRICE_POL_PREP_DEAL policy opcode.

By default, this policy opcode is an empty hook, but you can customize it to enhance **/deal** objects with additional data not provided either by the GUI application or by the Price List FM or to perform other custom actions.

This policy opcode is called before the Price List FM performs any validation checks.
- To enable validation during **/deal** object creation, use the PCM_OP_PRICE_POL_VALID_DEAL policy opcode. This policy opcode can be used to perform validations in addition to those performed by the Price List FM.

This policy opcode is called by PCM_OP_PRICE_COMMIT_DEAL. By default, this policy opcode is an empty hook; it returns the result PIN_PRICE_VERIFY_PASSED.
- To verify that deleting a **/deal** object is permitted, use the PCM_OP_PRICE_POL_DELETE_DEAL policy opcode. Use this opcode to perform validations in addition to those performed by the Price List FM.

PCM_OP_PRICE_COMMIT_DEAL calls this policy opcode before deleting the **/deal** object. By default, this policy opcode is an empty hook provided to facilitate customization. It returns the result PIN_PRICE_VERIFY_PASSED.

Success or failure is indicated by the value returned for PIN_FLD_RESULTS. When successful, PIN_FLD_RESULTS is set to **1**. When any operation fails, PIN_FLD_RESULTS is set to **0**.

Managing /plan Objects

Use the PCM_OP_PRICE_COMMIT_PLAN opcode to create, modify, or delete **/plan** objects in the BRM database.

This opcode accepts an flist consisting of a PIN_FLD_PLAN array, each element of which represents a **/plan** object.

PCM_OP_PRICE_COMMIT_PLAN determines whether to create, modify, or delete the specified **/plan** object:

- When the object does not exist, this opcode *creates* the specified **/plan** object by doing the following:
 - Calling the PCM_OP_PRICE_POL_PREP_DISCOUNT policy opcode to perform any custom preparation. By default, this policy opcode is an empty hook, but you can customize it to enhance **/plan** objects with additional data or to perform other custom actions.
 - Calling the PCM_OP_PRICE_POL_VALID_DISCOUNT policy opcode to perform any custom validation. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.
 - Creating the specified object in the BRM database.
- When the object already exists, this opcode *modifies* the specified **/plan** object in the BRM database.

Important: This opcode overwrites data in existing **/plan** objects, so be sure you pass in the correct object to modify.

- When the PIN_FLD_DELETED_FLAG is set, this opcode *deletes* the specified **/plan** object by doing the following:
 - Calling the PCM_OP_PRICE_POL_DELETE_DISCOUNT policy opcode to perform any custom validation. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.
 - Deleting the specified object in the BRM database.

PCM_OP_PRICE_COMMIT_PLAN performs all operations within a single transaction, so success or failure is indicated by the value returned for PIN_FLD_RESULTS:

- When successful, PCM_OP_PRICE_COMMIT_PLAN sets PIN_FLD_RESULTS to **1**.
- When any operation fails, PCM_OP_PRICE_COMMIT_PLAN_LIST sets PIN_FLD_RESULTS to **0**. It also returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

Managing /dependency Objects

Use the PCM_OP_PRICE_COMMIT_DEPENDENCY opcode to create, modify, or delete a **/dependency** object. This object specifies any dependencies between two discounts or between a discount and a plan.

This opcode accepts an flist that includes a PIN_FLD_DEPENDENCY array, each element of which represent a **/dependency** object.

Flist /dependency Arrays

The arrays defining prerequisite and mutually exclusive relationships require the names of the two product objects. The name of the product that *requires* another product is listed in the PIN_FLD_DEPENDENT_NAME field; the name of the *required* product is listed in the PIN_FLD_DEPENDEE_NAME field. For example, if your business has an email product that requires a Basic Dialup product, **email** is the dependee and **Basic Dialup** is the dependent:

```
0 PIN_FLD_DEPENDENCIES    ARRAY [12]
1 PIN_FLD_DEPENDEE_OBJ    POID [0] 0.0.0.1 /deal -1 0
1 PIN_FLD_DEPENDENT_OBJ   POID [0] 0.0.0.1 /deal -1 0
1 PIN_FLD_DEPENDEE_NAME   STR [0] "Basic Dialup"
1 PIN_FLD_DEPENDENT_NAME  STR [0] "email"
1 PIN_FLD_TYPE            ENUM [0] 1
```

PCM_OP_PRICE_COMMIT_DEPENDENCY performs all operations within a single transaction, so success or failure is indicated by the value returned for PIN_FLD_RESULTS:

- When successful, PCM_OP_PRICE_COMMIT_DEPENDENCY sets PIN_FLD_RESULTS to **1**.
- When any operation fails, PCM_OP_PRICE_COMMIT_DEPENDENCY sets PIN_FLD_RESULTS to **0**. It also returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

Customizing How to Create and Delete Dependencies

To customize how to create and delete dependencies, use the following policy opcodes:

- To enable data modification during **/dependency** object creation, use the PCM_OP_PRICE_POL_PREP_DEPENDENCY policy opcode. Use this policy opcode to enhance **/dependency** objects with additional data not provided by either the GUI application or by the Price List FM. For example, you would modify this policy opcode if your business requires that you change a mutually exclusive relationship into a required one, without using Pricing Center.

PCM_OP_PRICE_COMMIT_DEPENDENCY calls this policy opcode before creating a new **/dependency** object. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.

This policy opcode is called before the Price List FM performs any validation checks.

Success or failure is indicated by the value returned for PIN_FLD_RESULTS. When successful, PIN_FLD_RESULTS is set to **1**. When any operation fails, PIN_FLD_RESULTS is set to **0**.

- To enable validation during **/dependency** object creation, use the PCM_OP_PRICE_POL_VALID_DEPENDENCY policy opcode. This policy opcode can be used to change **/dependency** relationships without using Pricing Center.

This policy opcode is called by PCM_OP_PRICE_SET_PRICE_LIST and PCM_OP_PRICE_COMMIT_DEPENDENCY before creating a **/dependency** object. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.

Success or failure is indicated by the value returned for PIN_FLD_RESULTS. When successful, PIN_FLD_RESULTS is set to **1**. When any operation fails, PIN_FLD_RESULTS is set to **0**.

- To verify that deleting a **/dependency** object is permitted, use the PCM_OP_PRICE_POL_DELETE_DEPENDENCY policy opcode. Use this policy opcode to perform validations in addition to those performed by the Price List FM.

For example, you can use this policy opcode to confirm that the valid time period for the **/dependency** object has expired before allowing it to be deleted.

This policy opcode is called by PCM_OP_PRICE_COMMIT_DEPENDENCY before it deletes a **/dependency** object. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.

Success or failure is indicated by the value returned for PIN_FLD_RESULTS. When successful, PIN_FLD_RESULTS is set to **1**. When any operation fails, PIN_FLD_RESULTS is set to **0**.

Managing /transition Objects

Use the PCM_OP_PRICE_COMMIT_TRANSITION opcode to create, modify, or delete a **/transition** object.

This opcode accepts an flist consisting of a PIN_FLD_TRANSITION array, each element of which represents a **/transition** object.

PCM_OP_PRICE_COMMIT_TRANSITION determines whether to create, modify, or delete the specified **/transition** object:

- When the object does not exist, this opcode *creates* the specified **/transition** object by doing the following:
 - Calling the PCM_OP_PRICE_POL_PREP_TRANSITION policy opcode to perform any custom preparation. By default, this policy opcode is an empty hook, but you can customize it to enhance **/transition** objects with additional data or to perform other custom actions.
 - Calling the PCM_OP_PRICE_POL_VALID_DISCOUNT policy opcode to perform any custom validation. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.
 - Creating the specified object in the BRM database.
- When the object already exists, this opcode *modifies* the specified **/transition** object in the BRM database.

Important: This opcode overwrites data in existing **/transition** objects, so be sure you pass in the correct object to modify.

- When the PIN_FLD_DELETED_FLAG is set, this opcode *deletes* the specified **/transition** object by doing the following:
 - Calling the PCM_OP_PRICE_POL_DELETE_TRANSITION policy opcode to perform any custom validation. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.
 - Deleting the specified object in the BRM database.

PCM_OP_PRICE_COMMIT_TRANSITION performs all operations within a single transaction, so success or failure is indicated by the value returned for PIN_FLD_RESULTS:

- When successful, PCM_OP_PRICE_COMMIT_TRANSITION sets PIN_FLD_RESULTS to **1**.
- When any operation fails, PCM_OP_PRICE_COMMIT_TRANSITION sets PIN_FLD_RESULTS to **0**. It also returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

Customizing How to Create and Delete Transitions

To customize how to create and delete transitions, use the following policy opcodes:

- To enable data modification during **/transition** object creation, use the PCM_OP_PRICE_POL_PREP_TRANSITION policy opcode. Use this policy opcode to enhance **/transition** objects with additional data not provided by either the GUI application or by the Price List FM.

For example, if your business waives purchase and cancellation fees for its San Francisco customers, you would implement that logic in this policy opcode.

PCM_OP_PRICE_COMMIT_TRANSITION calls this policy opcode before creating a new **/transition** object. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.

This policy opcode is called before the Price List FM performs any validation checks.

Success or failure is indicated by the value returned for PIN_FLD_RESULTS. When successful, PIN_FLD_RESULTS is set to **1**. When any operation fails, PIN_FLD_RESULTS is set to **0**.

- To enable validation during **/transition** object creation, use the PCM_OP_PRICE_POL_VALID_TRANSITION policy opcode. This policy opcode can be used to change **/transition** relationships without using Pricing Center.

For example, if your business requires that you prohibit underage customers from transitioning to a specific deal, you would add that limitation to this policy opcode.

This policy opcode is called by PCM_OP_PRICE_COMMIT_TRANSITION before creating a **/transition** object. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.

This policy opcode is called before the Price List FM creates the **/transition** object.

Success or failure is indicated by the value returned for PIN_FLD_RESULTS. When successful, PIN_FLD_RESULTS is set to **1**. When any operation fails, PIN_FLD_RESULTS is set to **0**.

- To verify that deleting a **/transition** object is permitted, use the PCM_OP_PRICE_POL_DELETE_TRANSITION policy opcode. Use this policy opcode to perform validations in addition to those performed by the Price List FM.

For example, you can use this policy opcode to confirm that the valid time period for the **/transition** object has expired before allowing it to be deleted.

This policy opcode is called by PCM_OP_PRICE_COMMIT_TRANSITION before it deletes a **/transition** object. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.

Success or failure is indicated by the value returned for PIN_FLD_RESULTS. When successful, PIN_FLD_RESULTS is set to **1**. When any operation fails, PIN_FLD_RESULTS is set to **0**.

Managing /group/plan_list Objects

Use the PCM_OP_PRICE_COMMIT_PLAN_LIST opcode to commit **/group/plan_list** objects to the BRM database.

This opcode connects to the database, opens a transaction, and retrieves the **/plan_list** flist. PCM_OP_PRICE_COMMIT_PLAN_LIST searches the PIN_FLD_PLAN flist to match the name and type fields in the plan list flist. This plan is checked against existing plans in the database. If the plan does not exist, this opcode reports an error and stops the transaction. If the plan does exist, this opcode adds the flist to the PIN_FLD_RESULTS array. This process is repeated until all plan list entries are added to the PIN_FLD_RESULTS array and committed to the database. The committed changes include new, modified, or deleted objects.

PCM_OP_PRICE_COMMIT_PLAN_LIST performs all operations within a single transaction, so success or failure is indicated by the value returned for PIN_FLD_RESULTS:

- When successful, PCM_OP_PRICE_COMMIT_PLAN_LIST sets PIN_FLD_RESULTS to **1**.

- When any operation fails, PCM_OP_PRICE_COMMIT_PLAN_LIST sets PIN_FLD_RESULTS to 0. It also returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

Managing /sponsorship Objects

Use the PCM_OP_PRICE_COMMIT_SPONSORSHIP opcode to create, change, or delete a **/sponsorship** object.

PCM_OP_PRICE_COMMIT_SPONSORSHIP determines whether to create, modify, or delete the specified **/sponsorship** object:

- When the object does not exist, this opcode *creates* the specified **/sponsorship** object by doing the following:
 - Calling the PCM_OP_PRICE_POL_PREP_DISCOUNT policy opcode to perform any custom preparation. By default, this policy opcode is an empty hook, but you can customize it to enhance **/sponsorship** objects with additional data or to perform other custom actions.
 - Calling the PCM_OP_PRICE_POL_VALID_DISCOUNT policy opcode to perform any custom validation. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.
 - Creating the specified object in the BRM database.
 - Preparing a list of all **/sponsorship** objects that were created.
 - Generating the **/event/notification/price/sponsorships/modify** notification event.
- When the object already exists, this opcode *modifies* the specified object by doing the following:
 - Updating the specified object in the BRM database.
 - Preparing a list of all **/sponsorship** objects that were modified.
 - Generating the **/event/notification/price/sponsorships/modify** notification event.

Important: This opcode overwrites data in existing **/sponsorship** objects, so be sure you pass in the correct object to modify.

- When the PIN_FLD_DELETED_FLAG is set, this opcode *deletes* the specified **/sponsorship** object by doing the following:
 - Calling the PCM_OP_PRICE_POL_DELETE_DISCOUNT policy opcode to perform any custom validation. By default, this policy opcode is an empty hook, but you can customize it to validate field values or to perform other custom actions.
 - Deleting the specified object in the BRM database.

PCM_OP_PRICE_COMMIT_SPONSORSHIP performs all operations within a single transaction, so success or failure is indicated by the value returned in the PIN_FLD_RESULTS output field: a value of 1 indicates that all operations succeeded and a value of 0 indicates that the operation failed. If the operation fails, the opcode also

returns two additional fields: PIN_FLD_RESULT_FORMAT, which lists all error codes, and PIN_FLD_DESCR, which provides a concise error description.

Managing Filter Sets

Filter sets enable you to apply system products and system discounts to a select group of customers (for example, those with the best credit history). You define which customers and events qualify for specified products and discounts by using Pricing Center or a custom client application. Each filter set specifies:

- The criteria an EDR must meet to qualify for the filter set: that is, list of required EDR fields and their values
- The list of applicable system products and system discounts, along with their validity dates and priorities

BRM stores information about each filter set in **/filter_set/product** objects.

To manage filter sets, use the following opcodes:

- PCM_OP_FILTER_SET_CREATE. See ["Creating Filter Sets"](#).
- PCM_OP_FILTER_SET_UPDATE. See ["Updating Filter Sets"](#).
- PCM_OP_FILTER_SET_DELETE. See ["Deleting Filter Sets"](#).

Creating Filter Sets

Use the PCM_OP_FILTER_SET_CREATE opcode to create **/filter_set/product** objects in the BRM database. This opcode is called directly by Pricing Center.

PCM_OP_FILTER_SET_CREATE performs the following operations:

1. Confirms that the object to be created is a **/filter_set/product** object, or one subclassed from it. The type of object is specified in the PIN_FLD_POID field of the input flist.
2. Creates the **/filter_set/product** object.
3. Generates an **/event/filter_set/create** object to record details of the event.
4. Returns the POID of the new **/filter_set/product** object.

Updating Filter Sets

Use the PCM_OP_FILTER_SET_UPDATE opcode to modify existing **/filter_set/product** objects. This opcode is called directly by Pricing Center.

PCM_OP_FILTER_SET_UPDATE performs the following operations:

1. Determines whether to add, modify, or delete data in the object by checking the PIN_FLD_FLAG field:
 - When the flag is set to **0**, the opcode *modifies* filter set data.
 - When the flag is set to **1**, the opcode *adds* filter set data.
 - When the flag is set to **2**, the opcode *deletes* filter set data.
2. Modifies data in the specified **/filter_set/product** object.
3. Creates an **/event/filter_set/update** object to record details of the event.
4. Returns the POID of the **/filter_set/product** object.

Deleting Filter Sets

Use the PCM_OP_FILTER_SET_DELETE opcode to delete **/filter_set/product** objects from the BRM database. This opcode is called directly by Pricing Center.

PCM_OP_FILTER_SET_DELETE performs the following operations:

1. Deletes the specified **/filter_set/product** object.
2. Generates an **/event/filter_set/delete** object to record details of the event.
3. Returns the POID of the **/filter_set/product** object.

Testing Your Price List

This chapter describes the procedure for testing an Oracle Communications Billing and Revenue Management (BRM) price list. Testing is an important step to take before using the price list for customers. For information on creating a price list, see "[Setting Up Price List Data](#)".

For example, you can:

- Test hourly usage rates by generating usage activity. If your price list specifies a usage rate of \$1 per hour, and you generate 10 hours of usage, the resulting invoice should show a balance impact of \$10.
- Test the billing impact of cycle-forward rates by advancing the date within the BRM system and then running billing.

To test a price list, perform some or all of the following tasks, depending on the types of rates you must test. This chapter describes each task in detail:

- [Creating Test Accounts](#)
- [Generating Test Account Activity](#)
- [Advancing the Date](#)
- [Generating a Test Invoice](#)

For an example, see "[Example Price List Test](#)".

About Using a Test Database

Before testing a price list, set up a test database that is separate from your production database. Because you simulate account activity and time changes when you test your price list, you would create erroneous events in a production BRM database that contains real customer accounts.

Tip: You can remove all price list elements in a database by using the `loadpricelist utility -p` parameter. See "[loadpricelist](#)".

Creating Test Accounts

Use Customer Center to create test accounts in your test database. Note the login name for each account you create, because you will need it later.

Important: To create accounts with your new plans, you must add them to a plan list. See "Controlling which plans and deals are available to customers" in *BRM Managing Customers*.

When selecting a payment method, you can select either Invoice or Credit Card. If you select Credit Card, also do the following:

- Ensure that the necessary Data Managers and applications are properly configured and running:
 - For Paymentech, **dm_fusa**, **answer_b**, and **answer_s** should be running. For more information, see "Configuring BRM-initiated payment processing" in *BRM Managing Customers*.
- You must use one of the following pairs of credit card numbers and expiration dates listed in [Table 8-1](#) for your test accounts:

Table 8-1 Example Credit Card Expiration Data

Credit Card Number	Expiration Date
4444 1111 2222 3333	0999
4101 0000 0000 0000	any expiration date

Tip: A simple cycle rate, such as a monthly fee for a service, should appear in an account's balance as soon as you create the account.

Generating Test Account Activity

Use the **sample_act** utility to generate activity for a test account. You use this utility to simulate a customer using a product or service without the product or service actually being available. You can then see if the activity changes the test account's balance as you expect.

Perform these tasks to generate activity with **sample_act**:

- [Running sample_act](#)
- [Checking Results of Running sample_act](#)

Running sample_act

1. Go to *BRM_Home/source/apps/sample*.
2. Edit the **pin.conf** file for the **cm_ptr** to point to your CM process. Also, edit the login type, login name, and password entries.
3. Run the **sample_act** utility for each account you want to include in your test.

The syntax for **sample_act** depends on the type of event you are simulating:

- **Session event:** A usage event, such as hours of Internet access. If the rate depends on the passage of time, use a session event.

For a session event, use this syntax:

```
sample_act -v -e session -l login -d duration -s servicename
```

- **Activity event:** An item or cycle fee that does not depend on time, such as a monthly fee.

For an activity event, use this syntax:

```
sample_act -v -e activity -l login -q quantity -s servicename
```

Following is a description of the most common **sample_act** options.

- **-v:** Use to run **sample_act** in verbose mode. In this mode, you see status messages at the command prompt, including messages that the event generated successfully or that an error occurred. There is no argument.
- **[-n RUM Name -q quantity [-u Unit]]**
- **[-b event time]** (as MM:DD:YY:HH:MM:SS or unixtime)
- **[-A ANI] [-D DNIS] [-Y call_type]:** ANI is the source telephone number and DNIS is the destination telephone number
- **[-X int_indicator] [-O origin_gw]**
- **[-N new Timezone ID String]**
- **[-I termserve_id] [-P termserve_port] [-T old timezone offset]**
- **-e activity | session [-c custom_subtype]:** Enter activity or session.
- **-l login:** Enter the login name for the test account and service for which you want to generate activity.
- **-q quantity:** For an activity event, enter the quantity purchased.
- **-d duration:** For a session event, enter the duration of the simulated event in seconds. The default is 3600 seconds, or one hour.
- **-s service_type -l service_login:** Enter the name of the service for which you are simulating the event, such as **/service/ip**.

The following example simulates using three hours of Internet service. The login is **pin_user** and the service is **/service/ip**.

```
% sample_act -v -e session -l pin_user -d 10800 -s /service/ip
```

To test a rate that changes at different times, such as an Internet service rate with peak and off-peak rates, you can advance the system date in conjunction with running **sample_act**. See "[Advancing the Date](#)".

Before advancing the time, you must edit the **sample_act** configuration file so that **sample_act** responds to the changed time. See "[Setting Up the pin_virtual_time Utility](#)".

Checking Results of Running **sample_act**

To get the results of your test after running **sample_act**:

1. Find the test accounts in Customer Center.
2. Click the **Balance** tab.
3. For detailed information about balance changes, use Event Browser.

Advancing the Date

To properly test a price list, you must simulate the passage of time. For example, you should test:

- If accounts are billed correctly. Advance the date to the next billing cycle, then run billing and generate an invoice.
- If a time-based discount takes effect correctly. If you have a deal that includes 90 days of free email service, for example, advance the date more than 90 days to see if the account starts being charged for the service.
- If folds are working correctly. If a product includes 10 free hours of Internet service per month, for example, advance the date by a month or more to ensure that unused free hours do not carry over.

The `pin_virtual_time` utility enables you to simulate changes in the BRM system time.

Note: If you use `pin_virtual_time` with a BRM system that includes Pipeline Manager, you must set the `VirtualTime` parameter for the `DAT_BalanceBatch` module to `True` to ensure that balances are calculated correctly.

Caution: The `pin_virtual_time` utility works only on a single computer. Typically, you set up a test BRM system on just one computer. If you run `pin_virtual_time` on a system that is distributed across multiple computers, you must carefully coordinate time changes on all the computers.

Perform the following tasks to advance the date:

- [Setting Up the `pin_virtual_time` Utility](#)
- [Running `pin_virtual_time`](#)
- [Stopping and Starting BRM](#)

Setting Up the `pin_virtual_time` Utility

1. Go to the `BRM_Home/sys/test` directory.
2. Run this command to create a file that `pin_virtual_time` requires, called `pin_virtual_time_file`:

```
pin_virtual_time -m 0 -f BRM_Home/bin/pin_virtual_time_file
```

Note: `BRM_Home/lib/pin_virtual_time_file` is the standard path and file name, but you can change it.

3. Add the following entry to the configuration file in the `BRM_Home/sys/test` directory:

```
- - pin_virtual_time pin_virtual_time_file
```

Replace `pin_virtual_time_file` with the path and name of the mapped file created in the previous step.

Adding the entry to the configuration file in `sys/test` enables you to run `pin_virtual_time` from that directory. The directory from which you run `pin_virtual_time` must contain a configuration file with the `pin_virtual_time` entry.

4. Add the entry in step 3 to the configuration file for each program you want to respond to an altered time.

To test your price list, edit the configuration files for at least the applications listed in [Table 8-2](#):

Table 8-2 Configuration Files for Testing Applications

Application	Configuration File Location
CM	<code>sys/cm/</code>
Oracle DM	<code>sys/dm_oracle/</code>
Billing and invoice utilities	<code>apps/pin_billd/</code>
<code>sample_act</code> utility	<code>source/apps/sample</code>

To ensure that all applications respond to `pin_virtual_time` changes, add the entry to all the configuration files in your test BRM system.

Entry example:

```
- - pin_virtual_time BRM_Home/bin/pin_virtual_time_file
```

Running pin_virtual_time

After you set up the `pin_virtual_time` utility, run `pin_virtual_time` to advance the BRM date and time.

Syntax for `pin_virtual_time`:

```
pin_virtual_time -m 2 MMDDHHMM[CC]YY.[SS]
```

For the string `MMDDHHMM[CC]YY.[SS]`, enter the date and time you want BRM to use in this format: month, date, hour, minute, year, and seconds. You must enter at least two digits for the year, but you can enter four. Seconds are optional.

For example, to set the date and time to 9/3/99 and 11:30, respectively:

```
% pin_virtual_time -m 2 090311301999.00
```

The command displays this message at the command prompt:

```
filename BRM_Home/lib/pin_virtual_time_file, mode 2, time: Fri Sept 03 11:30:00
1999
```

The time then advances normally from the new reset time.

Stopping and Starting BRM

After you run `pin_virtual_time`, you must stop and restart BRM to read in the new time:

1. Log on to the computer that is the server for your test BRM system. Usually, your test system should be on a single machine.

Note: Log in as something other than `root`.

2. Exit all BRM client tools, such as Customer Center and Pricing Center.
3. Stop and restart the CM and Oracle Data Manager.

Generating a Test Invoice

Generate invoices for your test accounts to see if the accounts are billed correctly. Before following this procedure, you should have generated account activity and advanced the date to the beginning of the next billing cycle. Go to the **Payment** tab in Customer Center to find the Next Billing Start date for an account.

To run billing and generate a test invoice:

1. Go to the *BRM_Home/apps/pin_bill* directory.
2. At the command prompt, enter the following:

```
% pin_bill_day
```

This command is a script that runs several billing utilities.

BRM creates an invoice for each active account.

3. Open a test account in Customer Center and go to the **Payment** tab.
4. Ensure that the Current Billing Start and End dates are what you expect.
5. Search for the invoice for the current account, and see that it contains the results of your test. See "Displaying invoices" in *BRM Designing and Generating Invoices*.

Example Price List Test

This example takes you through the steps for testing a plan that includes the following:

- A monthly Internet access fee of \$9.95.
- An hourly fee of \$2.00 for prime-time Internet service usage. Prime time is 8:00 a.m. to 6:00 p.m. weekdays.
- An hourly fee of \$1.00 for off-peak Internet service usage. This includes nights and weekends.

You can test this plan as follows:

1. In Customer Center, create a test account by using the plan you are testing.
2. Go to the **Balance** tab. The current balance for the US Dollar resource should be \$9.95, because the monthly access fee impacts the balance right away.
3. Close the test account.
4. Set up the **pin_virtual_time** utility, by using the procedure in "[Setting Up the pin_virtual_time Utility](#)". Ensure that you at least add the **pin_virtual_time** entry to the configuration files for the Connection Manager, Oracle Data Manager, and **sample_act** utility.
5. Go to the *BRM_Home/sys/test* directory.
6. Change the BRM system time so it is close to the time when one rate ends and the other begins. For example, set the time to 6:00 a.m. the following day.

If you created the account on May 1, 1999, you would run the following command to change BRM's system time to 6:00 a.m. the next morning:

```
pin_virtual_time -m 2 0502060099 -l pin_user
```

In the above command, `pin_user` is the login assigned to the test account.

7. Stop and restart BRM, by using the procedure in "[Stopping and Starting BRM](#)".
8. Go to *BRM_Home* `/source/apps/sample`.
9. Generate four hours of Internet access activity for the account by running this command:

```
sample_act -v -e session -r ip/dialup/async/hourly -s /service/ip -d 14400 -l pin_user
```

In the above command, `ip/dialup/async/hourly` is the rate category, `/service/ip` is the service, 14400 is four hours in seconds, and `pin_user` is the login name for the account.

10. Reopen the account in Customer Center.
11. Go to the **Balance** tab.

The current balance for the US Dollar resource should now be \$15.95. The `sample_act` command should have generated two hours of Internet access at the off-peak fee of \$1 and two hours at the prime-time fee of \$2. If you set up the rates correctly, this adds \$6.00 to the previous balance of \$9.95.

Using the XML Pricing Interface to Create a Price List

This chapter describes how to create and edit an Oracle Communications Billing and Revenue Management (BRM) price list by using the XML Pricing Interface.

Before reading this chapter, read "[About Creating a Price List](#)".

About Creating and Modifying Price List Files

You create and modify price lists by using the following:

- **Pricing Center:** Pricing Center is a GUI application that provides on-screen wizards to help you create your price list. You create and modify the price list and retrieve and save it to the database all directly in Pricing Center. For more information, see Pricing Center Help.
- **XML Pricing Interface:** The XML Pricing Interface consists of the "[loadpricelist](#)" utility. You create price lists directly in an XML editor or text editor and then use the utility to load, retrieve, and delete the price list from the BRM database. For information on creating price lists with the XML Pricing Interface, see "[About the XML Pricing Interface](#)".

About the XML Pricing Interface

The XML Pricing Interface is a command-line utility called **loadpricelist** that you can use to download, edit, and load price lists.

Important: The XML price list must follow the format detailed in the XML SchemaDefinition (XSD) file **price_list.xsd**. By default, the **price_list.xsd** file is installed in *BRM_Home/PricingCenter* and/or *BRM_Home/setup/scripts*.

About Downloading and Loading XML Price List Files

You can use the **loadpricelist** utility to download and load an entire price list or a partial price list. When you download a price list from the database, the utility writes the price list information you specify into an XML file. If you specify to download to an existing price list file, the utility overwrites the entire XML file.

When you load a price list from an XML file into the database, the utility modifies only those objects that have changed:

- If an object has changed, the utility updates the object's attributes in the database.

- If an object does not exist in the database, the utility creates the database object.
- If an object has not changed and is already in the database, the utility does not modify the database object.

Prerequisites for the XML Pricing Interface

Before you use the XML Pricing Interface, you must configure the following:

- Include the **loadpricelist.jar**, **xerces.jar**, **pcm.jar**, and **pcmext.jar** files in your CLASSPATH. The **xerces.jar**, **pcm.jar**, and **pcmext.jar** files are located in the **C:\Program Files\Common Files\Portal Software** directory and subdirectories by default.

Tip: Instead of including the JAR files in your CLASSPATH, you can include them in the **loadpricelist** file.

- Include the location of the **Infranet.properties** file in your CLASSPATH.
The default **Infranet.properties** file is in the **\common** directory. To put the **Infranet.properties** file in the local directory with the **loadpricelist** utility, add **./** (period-slash) to the beginning of the **Infranet.properties** CLASSPATH.
- Ensures that the **Infranet.properties** file specifies the correct login information for the BRM database you want to connect to.

Note: You might need to change this information if, for example, you want to download a price list from one BRM database, modify it, and load it to a different BRM database.

- Install Pricing Center or copy the **price_list.xsd** file to your system. You can find the **price_list.xsd** file on any BRM server in either the **BRM_Home/PricingCenter** directory or the **BRM_Home/setup/scripts** directory.

Starting the XML Pricing Interface

To start the XML Pricing Interface, at the command prompt, enter **loadpricelist** followed by an interactive or non-interactive command.

Getting Help for the XML Pricing Interface

To get help for the XML Pricing Interface:

1. Start the XML Pricing Interface, if necessary.
2. Enter **help** at the command prompt and press **Enter**.

Working in Interactive and Non-Interactive Mode

You can work in interactive and non-interactive mode in the XML Pricing Interface.

- In interactive mode, you issue a command for each action. For a list of commands that can be used in this mode, see "[Syntax: Interactive Mode](#)".
- In non-interactive mode, you can use special commands that batch several related actions together so the system performs them automatically. For a list of

parameters that can be used in this mode, see "[Syntax: Non-Interactive Mode](#)".

You receive a confirmation message when the XML Pricing Interface carries out a command successfully. If you do not receive a confirmation message, look in the utility log file (**javapcm.log**) to find any errors. The log file is either in the directory where the utility was started or in a directory specified in the **Infranet.properties** file.

Note: If an error occurs during a command, none of the command is carried out even if the error was only caused by part of the operation.

Creating Price Lists with the XML Pricing Interface

You create price lists in an XML editor or text editor and then load it into the BRM database by using the "[loadpricelist](#)" utility.

To create a price list:

1. Plan your price list. You should know in advance how you will set up your rates, products, deals, and plans. Ensure that the following pricing objects and data have been created in the BRM database:
 - services
 - events
 - resources
 - currency exchange rates
 - G/L IDs
 - tax codes and tax suppliers
 - ratable usage metrics (RUMs)
 - product-level provisioning categories
 - zones

For information on planning your price list, see "[About Creating a Price List](#)".

2. Create an XML file, such as **price_list.xml**, in a text editor or XML editor.
3. Enter your price list information. Ensure that your price list follows the structure defined in the **price_list.xsd** file.

Note: You can view sample XML price list files in the *BRM_Home\PricingCenter\Sample_Price_Plans* directory.

4. Save and close the file.
5. Ensure that any billable events in your price list are also included in the **pin_event_map** file. Otherwise, they will not be rated. See "[Mapping Event Types to Services](#)".
6. Load the price list into the BRM database. See "[Loading a Price List into the Database with the XML Pricing Interface](#)".

BRM can begin using the price list after it is committed to the database.

Modifying Existing Price Lists with the XML Pricing Interface

To modify an existing price list file:

1. Retrieve the existing price list file. The method you use depends on the file's location:
 - **The XML Pricing Interface:** Your price list file is saved in XML format. You can open it directly in an XML editor or text editor.
 - **The BRM database:** You must download the price list to your system and write the information to an XML file. See "[Downloading Price Lists from the Database with the XML Pricing Interface](#)".
2. Open the XML price list file in a text editor or XML editor.
3. Edit your price list file. Ensure that it follows the structure defined in the **price_list.xsd** file. By default, **price_list.xsd** is located in *BRM_Home\PricingCenter*.

Note: You can view sample XML price list files in the *BRM_Home\PricingCenter\Sample_Price_Plans* directory.

4. Save and close the file.
5. Ensure that any billable events in your price list are also included in the **pin_event_map** file. Otherwise, they will not be rated. See "[Mapping Event Types to Services](#)".
6. Load the price list into the BRM database. See "[Loading a Price List into the Database with the XML Pricing Interface](#)".

BRM can begin using the price list after it is committed to the database.

Moving a Price List between Databases with the XML Pricing Interface

You can use the **loadpricelist** utility to move a price list from one database to another (for example, from a test database to a production database). You move the price list by downloading it from a *source* BRM database and then loading it into a *destination* BRM database.

To move a price list from one database to another:

1. Download the price list from the source database. See "[Downloading Price Lists from the Database with the XML Pricing Interface](#)".
2. If necessary, edit the XML file in a text editor or XML editor.
3. Load the price list into the destination BRM database. See "[Loading a Price List into the Database with the XML Pricing Interface](#)".

Downloading Price Lists from the Database with the XML Pricing Interface

You can download either the entire price list or just a subset of the price list from the BRM database into an XML file that can be edited.

Note: Price lists are stored in the database in *flist* format. The utility converts the file into XML format as part of the download process.

Downloading a Full Price List

To download a full price list:

1. Ensure that the **Infranet.properties** file in your CLASSPATH has the correct login information for the BRM database you want to download price list information from. See "Setting global options" in *BRM Developer's Guide*.
2. Retrieve the price list from the BRM database and write it to an XML file.

Interactive mode

```
loadpricelist
retrieve
write [PriceListFile]
```

where *PriceListFile* specifies the name and location of the file to which to extract the data.

Caution: You can retrieve only one price list at a time. If you retrieve another price list when one is already in internal memory, the existing internal list is overwritten.

Non-interactive mode

```
loadpricelist [-f] -r PriceListFile
```

where *PriceListFile* specifies the name and location of the file to which to extract the data.

Your price list is written to the specified XML file. You can now edit it directly in an XML editor.

Downloading a Subset of the Price List

You can write specific pricing objects, such as **/product**, **/discount**, or **/sponsorship** objects, rather than the entire price list to the XML price list file.

To download a subset of the price list:

1. Ensure that the **Infranet.properties** file in your CLASSPATH has the correct login information for the BRM database you want to connect to.
2. Specify the pricing objects to retrieve according to their service type, modification time, or object type, and write them to a specified XML file.

Interactive mode

```
loadpricelist
retrieve [-s ServiceType] [-t ModifiedTime] [product] [discount] [sponsorship]
write PriceListFile
```

where:

- **-s ServiceType** specifies to retrieve the specified service type. You can list multiple service types by using a comma (,) as a delimiter.

Note: Do not use this parameter with the **sponsorship** parameter.

- **-t *ModifiedTime*** retrieves objects that were modified after the specified timestamp.
The time is specified in the ISO-8601 format: *YYYY-MM-DDThh:mm:ssTZD*, with the server's local time zone as the default. For example:
`-t 1997-07-16T19:20:30+01:00`
- **product** specifies to retrieve only the **/product** objects from the database.
- **discount** specifies to retrieve only the **/discount** objects from the database.
- **sponsorship** specifies to retrieve only the **/sponsorship** objects from the database.
- *PriceListFile* specifies the name and location of the file to which to extract the data.

Non-interactive mode

```
loadpricelist [-f] -r PriceListFile [-P][-D][-S][-s ServiceType] [-t ModifiedTime]
```

where:

- **f** specifies to execute the command without prompting for a confirmation.
- *PriceListFile* specifies the name and location of the file to which to extract the data.
- **-P** specifies to retrieve only the **/product** objects from the database.
- **-D** specifies to retrieve only the **/discount** objects from the database.
- **-S** specifies to retrieve only the **/sponsorship** objects from the database.
- **-s *ServiceType*** retrieves objects based on the specified service type. You can list multiple service types by using a comma (,) as a delimiter.

Note: Do not use this parameter with the **-S** parameter.

- **-t *ModifiedTime*** retrieves objects that were modified after the specified timestamp.
The time is specified in the ISO-8601 format: *YYYY-MM-DDThh:mm:ssTZD*, with the server's local time zone as the default. For example:
`-t 1997-07-16T19:20:30+01:00`

Downloading Price Lists in Batches

If you have a large price list, downloading it by using **loadpricelist** can require more than the available system memory. In this situation, you should configure **loadpricelist** to retrieve and write pricelists in batches. When configured this way, **loadpricelist** performs a step search and writes the results into multiple XML files based on the batch size you specify.

You use the **infranet.batchsize** entry in the **loadpricelist Infranet.properties** file to configure the batch size. The batch size is expressed as the number of pricing objects per batch. For example, the following entry configures **loadpricelist** to retrieve and write pricing objects in batches of 500.

```
infranet.batchsize=500
```

When a batch size is configured, **loadpricelist** creates files based on the file name you specify in the command and on the type of pricing object, incrementing the name appropriately for each file created. For example, if the batch size is set to 500 and there are 1900 pricing objects in the database, then **loadpricelist** creates four XML files, the first three with 500 pricing objects and one with the remaining 400 objects.

When retrieving pricing objects in batches, you cannot choose which type of pricing object to retrieve; for example, only products, or only discounts. The **-P**, **-D**, **-S**, **-s**, and **-t** parameters are ignored.

In this scenario, this command:

```
loadpricelist -rf MyPriceList
```

results in the following four files being created (along with others for the other pricing objects in the database):

```
MyPriceList_product_1.xml
```

```
MyPriceList_product_2.xml
```

```
MyPriceList_product_3.xml
```

```
MyPriceList_product_4.xml
```

File naming follows the same pattern for all object types. For example, file names for **/discount** objects would start with **MyPriceList_discount_1.xml** and increment from there.

The **batchsize** entry is not included in the **loadpricelist** file by default. To set the batch size:

1. Open the **Infranet.properties** file for **loadpricelist** in a text editor.

The default **Infranet.properties** file is in the **\common** directory, but may be located in the local directory with the **loadpricelist** utility.

2. Add the **batchsize** entry to the file using the following syntax:

```
infranet.batchsize=batch_size
```

where *batch_size* represents the number of pricing objects to include in each batch.

3. Save and close the file.

Loading a Price List into the Database with the XML Pricing Interface

After your price list is complete, you load it into the BRM database. The XML Pricing Interface automatically converts the XML price list into flist format when it is loaded into the database.

Important: The price list file must be in the same directory as the **price_list.xsd** file. If they are not in the same directory, edit the **noNamespaceSchemaLocation** entry in the price list file to include the full path to **price_list.xsd**. By default, **price_list.xsd** is in the *BRM_Home\PricingCenter* directory. For example, if the XSD file is located in **C:\pricelists**, change the entry to:
xsi:noNamespaceSchemaLocation="C:\pricelists\price_list.xsd"

To load a price list:

1. Ensure that the XML price list file is complete and follows the guidelines specified in the XSD file.
2. Ensure that the **Infranet.properties** file in your CLASSPATH has the correct login information for the BRM database you want to connect to.
3. Commit the price list XML file to the BRM database.

Interactive mode

```
loadpricelist
read PriceListFile
commit
```

where *PriceListFile* specifies the name and location of the file to load into the database.

Non-interactive mode

```
loadpricelist [-f] -c PriceListFile
```

where:

- *f* specifies to execute the command without prompting for a confirmation.
- *PriceListFile* specifies the name and location of the file to load into the database.

Purging Price Lists from the BRM Database

You can purge the following from the BRM database:

- All price list objects. See ["Purging the Entire Price List from the BRM Database"](#).
- A subset of the price list, based on the object type. See ["Deleting a Subset of the Price List from the BRM Database"](#).
- A specific price list object. See ["Deleting Pricing Objects from the BRM Database"](#).
- All dependency or transition objects. See ["Purging Dependency or Transition Objects from the BRM Database"](#).

Purging the Entire Price List from the BRM Database

To purge the entire price list from the database:

1. Ensure that the **Infranet.properties** file has the correct login information for the BRM database.
2. Purge the price list from the BRM database:

Interactive mode

```
loadpricelist
purge
```

Non-interactive mode

```
loadpricelist [-f] -p
```

where *f* specifies to execute the command without prompting for a confirmation.

Deleting a Subset of the Price List from the BRM Database

Important: To delete a subset of price list data from the BRM database, you must use interactive mode.

To delete a subset of the price list:

1. Ensure that the **Infranet.properties** file has the correct login information for the BRM database.
2. Start **loadpricelist** in interactive mode and read *PriceListFile* into internal memory.

```
loadpricelist
read PriceListFile
```

3. Specify the pricing data to delete:

```
delete [planlist[new|addon] | plan | bestpricing | deal | product | discount |
sponsorship]
```

where:

- **planlist [new|addon]** specifies to delete the plan list information from the BRM database. You can optionally specify to delete only the **new** plan lists or the **addon** plan lists.
- **plan** specifies to delete only the **/bestpricing** objects from the BRM database.
- **deal** specifies to delete only the **/deal** objects from the BRM database.
- **product** specifies to delete only the **/product** objects from the BRM database.
- **discount** specifies to delete only the **/discount** objects from the BRM database.
- **sponsorship** specifies to delete only the **/sponsorship** objects from the BRM database.

Deleting Pricing Objects from the BRM Database

Important: To delete individual pricing objects from the BRM database, you must use interactive mode.

To delete pricing objects:

1. Ensure that the **Infranet.properties** file has the correct login information for the BRM database.
2. Start **loadpricelist** in interactive mode and read *PriceListFile* into internal memory.

```
loadpricelist
read PriceListFile
```

3. Specify the pricing object to delete.

```
delete PriceObject
```

where *PriceObject* specifies the pricing object to delete.

Purging Dependency or Transition Objects from the BRM Database

Important: To delete only dependency or transition objects from the BRM database, you must use interactive mode.

To delete dependency or transition objects:

1. Ensure that the **Infranet.properties** file has the correct login information for the BRM database.
2. Specify the objects to delete.
 - To delete only the dependency objects, enter:

```
loadpricelist  
delete dependency
```
 - To delete only the transition objects, enter:

```
loadpricelist  
delete transition
```

Migrating Price List Data from Legacy Databases

This chapter describes the process of migrating legacy pricing and discounting data to the Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager database. Before you read this chapter, you should be familiar with database administration tasks. In addition:

- You must have in-depth knowledge of your legacy database.
- You must have in-depth knowledge about the Pipeline Manager database.
- You must know XML programming.
- You must have a good understanding of XSD. See ["Working with and Modifying the XSD"](#).
- You must understand the objects that are supported by the Pipeline Manager database and know the order in which the objects must be loaded into the Pipeline Manager database. For information, see ["About the Types of Objects to Migrate"](#).

Important: Pipeline Configuration Manager is an optional component.

About Migrating Legacy Data

The Pipeline Configuration Manager is a set of scripts and utilities you can use to load data into the Pipeline Manager database.

Use the **LoadIfwConfig** utility to load pricing data into the Pipeline Manager database. You can migrate legacy data by adding the data to an XML file and running the **LoadIfwConfig** utility to load the data.

Any type of legacy database such as Oracle can be migrated to the Pipeline Manager database by using the **LoadIfwConfig** utility.

You use the **LoadIfwConfig** utility to migrate each table of your legacy database to the Pipeline Manager database. For more information, see ["Loading Legacy Data into the Pipeline Manager Database with LoadIfwConfig"](#).

During data migration, if the legacy system and Pipeline Manager are run in parallel, keep the databases synchronized by migrating data every time you make a change in the legacy database.

Overview of the Migration Process

The migration process follows these steps:

1. Planning the migration.
This step includes comparing the data in the legacy database and the Pipeline Manager database and figuring out how to migrate the data from one database structure to another. See ["Guidelines for Mapping Legacy Data"](#).
2. Extracting legacy data to an XML file by using a data transformation program. See ["About Migrating Legacy Data"](#).
3. Loading data into the Pipeline Manager database by using the **LoadIfwConfig** utility. See ["Loading Legacy Data into the Pipeline Manager Database with LoadIfwConfig"](#).

Legacy Data XML File Example

Legacy data representation in the XML file

```
<?xml version="1.0" encoding="UTF-8"?>
<IFW xmlns="http://www.portal.com/tools/pricing"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.portal.com/tools/pricing IfwConfig.xsd">
<UOM UnitsOfMeasurement="hp1" Name="abc"/></IFW>
```

Guidelines for Mapping Legacy Data

You map the legacy pricing data to the pipeline pricing data in an XML file. This XML structure is defined in the XSD (**IfwConfig.xsd**). See ["Working with and Modifying the XSD"](#).

The structure of legacy pricing data is almost always different from the structure used by Pipeline Manager. For example, the legacy data might have pricing element equivalent to a pipeline rate plan, but not the equivalent of a pipeline rate plan version.

When you map legacy data, compare the structure of your legacy data against the structure of the Pipeline Manager data. [Table 10–1](#) shows a sample comparison:

Table 10–1 Comparing Legacy and Pipeline Manager Data

Legacy Data	Pipeline Manager Data
Rate plan	Rate plan
<i>No equivalent</i>	Rate plan version
Rate plan configuration	Rate plan configuration
Rate plan model	<i>No equivalent</i>

In this example:

- You must create rate plan versions. This data might be in another part of the legacy data; for example, the version information might be included in the legacy rate plan itself.
- You must map the data in the legacy rate plan model to one of the pipeline rate plan elements.

- The legacy data and the Pipeline Manager data have rate plans and rate plan configuration. However, you must ensure that those elements include the same data.
- The XML file you create must use the structure defined in the XSD.

If you cannot directly map the data, then export the legacy data to flat files, and run a data transformation program on the flat files to produce data that can be loaded into the Pipeline Manager database. You can use any third-party tool for this purpose, depending on the type of legacy database. XML is a widely accepted programming language so many parsers and tools are available to transform data to XML.

Rate Plan XML File Example

The following examples show the rate plan structure from the XSD file, and a rate plan in the XML file:

Rate plan definition in the XSD file

```
<xs:element name="RatePlan">
  <xs:annotation>
    <xs:documentation>...</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:attribute name="RatePlanCode" type="xs:string" use="required"/>
    <xs:attribute name="Name" type="xs:string"/>
    <xs:attribute name="Status" default="D">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:length value="1"/>
          <xs:minLength value="1"/>
          <xs:maxLength value="1"/>
          <xs:enumeration value="S"/>
          <xs:enumeration value="T"/>
          <xs:enumeration value="A"/>
          <xs:enumeration value="D"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="SystemBrandCode" type="xs:string"/>
    <xs:attribute name="CurrencyCode" type="xs:string" use="required"
fixed="USD"/>
    <xs:attribute name="IsSplitting" type="xs:boolean" default="0"/>
    <xs:attribute name="Type" default="R">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:length value="1"/>
          <xs:minLength value="1"/>
          <xs:maxLength value="1"/>
          <xs:enumeration value="W"/>
          <xs:enumeration value="R"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="TaxTreatment" type="xs:boolean" use="required"/>
    <xs:attribute name="CalendarCode" type="xs:string" use="required"/>
    <xs:attribute name="UTCTimeOffset" type="xs:string" default="+0100"/>
  </xs:complexType>
</xs:element>
```

Rate plan representation in the XML file

```

<RatePlan
RatePlanCode="ratepc"
Name="rateplanc"
Status="D"
SystemBrandCode=" "
CurrencyCode="USD"
IsSplitting="0"
Type="R"
TaxTreatment="1"
CalendarCode="cal2001"
UTCTimeOffset="+0100" />

```

Mapping Legacy Data

You extract the legacy data to a file by using a data conversion program. After the data is extracted from the legacy database, you migrate the data into an XML file. (You can also extract the data directly into an XML file.) You can insert different types of data, objects, and attributes in a single XML file.

To map legacy data, create an XML file that uses the structure defined in the XSD. Then migrate the objects in the specified order. See "[About the Types of Objects to Migrate](#)".

About the Types of Objects to Migrate

Important: For information about the objects and fields in the Pipeline Manager database, see the documentation in *Pipeline_home/database*.

The objects that can be migrated to the Pipeline Manager database are defined in the XSD.

[Table 10–2](#) shows the objects that can be migrated to the Pipeline Manager database and the prerequisite objects required for some objects. The prerequisite objects are the parent objects, which have objects dependent on them.

You must load child objects in the order shown in the table, as data in some objects depends on the existence of other objects. You load parent objects before child objects. However, sometimes a parent object can be loaded after loading a child object if the field in the child object that relates to parent object is not a mandatory field. For example, the **Geomodel** object is a parent of the **Zonemodel** object. However, the **Geomodel** field in the **Zonemodel** object is not a mandatory field, so you can load the **Zonemodel** object first.

Table 10–2 Order to Be Used in Loading Child Objects

Objects	Parent Object Prerequisites
Edrcdesc	None
Edrcfield	Edrcdesc
Aliasmap	Edrcfield Edrcdesc
Calendar	None
Holiday	Calendar

Table 10–2 (Cont.) Order to Be Used in Loading Child Objects

Objects	Parent Object Prerequisites
Pipeline	Edrcdesc
Uom	None
Currency	None
Resource	Currency
Taxcode	None
Taxgroup	None
Tax	Taxgroup Taxcode
GLaccount	Taxcode
Revenuegroup	None
Rum	None
Rumgroup	None
Rumgrouplnk	Rumgroup Rum Uom
Service	GLaccount Revenuegroup Rumgroup
Serviceclass	Service
Refmap	None
Mapgroup	None
Servicemap	Mapgroup Service
Usageclass	None
Usageclassmap	Usageclass Mapgroup
Uscgroup	None
Rscgroup	None
Apngroup	None
Zonemodel	Geomodel Apngroup
Impactcat	None
Geomodel	Ruleset
Geoarealnk	Geomodel
Geozone	Zonemodel Service Impactcat

Table 10–2 (Cont.) Order to Be Used in Loading Child Objects

Objects	Parent Object Prerequisites
Standardzone	Service Zonemodel Impactcat
Usagetype	None
Uscmap	Usagetype Zonemodel Uscgroup Impactcat
Apnmap	Apngroup Impactcat
Discarding	Pipeline
Daycode	None
Timeinterval	None
Timezone	None
Timemodel	None
Timemodellnk	Timezone Daycode Timeinterval Timemodel
Pricemodel	None
Pricemdlstep	Pricemodel Resource Rum GLaccount Revenuegroup
Discountmodel	None
Dscmdlver	Discountmodel
Discountmaster	None
Discountdetail	Discountmaster
Dsctrigger	None
Dscondition	Dsctrigger
Discountrule	Discountmaster
Discountstep	Discountrule
Dscmdlcnf	Discountrule Dsctrigger Dscmdlver
Rateplan	Calendar Systembrand Currency

Table 10–2 (Cont.) Order to Be Used in Loading Child Objects

Objects	Parent Object Prerequisites
Rateplanver	Zonemodel Rateplan
Rateplancnf	Service Rateplanver Pricemodel Timezone Timemodel Impactcat
Rateadjust	Rateplanver
Specialdayrate	None
Specialdaylnk	Rateplanver Specialdayrate
Scenario	Edrcdesc
Uommap	Uom Rum
Systembrand	None
Exchangerate	Currency
Rscmap	Serviceclass Rscgroup
Rule	None
Ruleitem	None
Ruleset	None
Rulesetlist	Ruleset
Sla	Uscgroup Rscgroup Ruleset
Splittingtype	Pipeline Systembrand
IcDaily	None
IcDailyalternate	None
Condition	Scenario Edrcfield
Icproduct	None
Networkoper	Taxgroup Currency
Networkmodel	Rateplan Systembrand Networkoper Currency

Table 10–2 (Cont.) Order to Be Used in Loading Child Objects

Objects	Parent Object Prerequisites
Icproductgrp	Networkmodel
Icproductcnf	Icproduct Icproductgrp
Icproductrate	Icproduct Networkkoper Rateplan Networkmodel
Noproduct	Networkkoper Currency
Noproductcnf	ICproduct Timezone Noproduct Impactcat
Nosp	Mapgroup
Poi	None
Segment	None
Segratelnk	Rateplan Segment
Segzonalnk	Segment Zonemodel
Ciberocc	Networkkoper
Socialnumber	None
Seqlogout	None
Dbversion	None
Iscript	None
Tam	None
Seqcheck	None
SeqlogIn	None
Duplicatecheck	None
Csstate	None
Destindesc	None
Classitem	None
Class	None
Queue	None
Changeset	None
Switch	Networkkoper
Trunk	Switch Networkkoper

Table 10–2 (Cont.) Order to Be Used in Loading Child Objects

Objects	Parent Object Prerequisites
Grouping	Scenario Edrcfield
Groupingcnf	Grouping Class
Classcon	GroupingCnf
Poiarealnk	Networkmodel Poi
Nobillrun	Networkoper Networkmodel
Aggregation	Scenario Edrcfield
Classlnk	Class Classitem
Classconlnk	Classitem Classcon
Dictionary	Queue
Trunkcnf	Trunk IcProductgrp Networkmodel Poi Switch Networkoper
Lergdata	Geoarealnk
Dscbalimpact	Discountstep
Cslock	Changeset
Csaudit	Changeset
Csreference	Changeset

Working with and Modifying the XSD

The XSD describes the structure of the XML document. The XML file you create must comply with the structure defined in the XSD.

The XSD defines the following items for an XML file:

- The elements and attributes, their data types, and the default and fixed values for the elements and attributes.
- Elements that are child elements, and the number and order of the child elements.
- If an element can be empty or can include text.

You can also modify the XSD to provide default values for certain attributes such as status, and provide domain constraints to the attributes. For example, the status can be A=active, D=de-active, T=test, or S=simulation. If the status S is not valid, you can restrict the XSD to allow only the status A, D, or T.

Note: The structure of the default XSD cannot be changed.

Before Loading Legacy Data

Before you begin to load the legacy data, you must do the following:

- Install and configure the Pipeline Manager database.
- Prepare the legacy data for mapping. See "[Guidelines for Mapping Legacy Data](#)".
- Map the legacy data to Pipeline Manager database format. See "[Mapping Legacy Data](#)".

Configuring the Registry File for LoadIfwConfig Utility

The **LoadIfwConfig.reg** file provides database connection information to the **LoadIfwConfig** utility. You can edit this file manually, but it is also updated when you run the **pin_setup** utility.

The **LoadIfwConfig.reg** file is located in *Pipeline_home/tools/XMLloader*.

Most of the entries are standard connection entries, with these exceptions:

- Use the **LogFileName** entry to specify the file where debug messages are written.

Important: To record debug messages, you must use the **verbose on** command when you run the **LoadIfwConfig** utility.

- Use the **LoadDataFromDB** entry to increase performance. Enabling this entry loads all the price list data from the database into memory, where the utility can access it faster.

Sample **LoadIfwConfig.reg** file:

```
LOADIFWCONFIG
{
  DataPool
  {
    Database
    {
      ModuleName = DbInterface
      Module
      {
        DatabaseName = dduttadb
        UserName      = PIN
        Password      = 5A46BAEBC6C2C1C3A796C20A000E1E091066017D
        AccessLib     = oci61
      }
    }
  }
  XMLSchemaFile = Metadata.xml
  LogFileName   = LoadIfwConfig.log
  LoadDataFromDB = False
}
```

Loading Legacy Data into the Pipeline Manager Database with LoadIfwConfig

To load legacy data into the Pipeline Manager database:

1. Go to *Pipeline_home/bin*.
2. Run **LoadIfwConfig** using one of the following commands:

- **Non-interactive mode**

```
LoadIfwConfig [-i] input_file [-I] [-c]
```

- **Interactive mode**

```
LoadIfwConfig [read input_file] [Insert] [commit]
```

Important: The utility must be run from the directory where the XML file is located.

The **LoadIfwConfig** utility uploads the data from the XML file into the Pipeline Manager database and commits the data. If there is failure, the **LoadIfwConfig** utility rolls back the data and notifies you on your computer screen.

Deleting Data from the Pipeline Manager Database with LoadIfwConfig

To delete an object in a table, you must specify the primary field of the object in the input XML file. The object can be deleted only if the primary field has no dependent child objects.

To delete data from the Pipeline Manager database:

1. Go to *Pipeline_home/bin*.
2. Run **LoadIfwConfig** using one of the following commands:

- **Non-interactive mode**

```
LoadIfwConfig [-i] input_file [-p] [-c] output_file
```

- **Interactive mode**

```
LoadIfwConfig [read input_file] [delete] [commit] output_file
```

Only the dependent objects are deleted, and the deleted objects are backed up in the output file.

Deleting Data Sample XML File

Deleting data representation in the XML file

```
<?xml version="1.0" encoding="UTF-8"?>
<IFW xmlns="http://www.portal.com/tools/pricing"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.portal.com/tools/pricing IfwConfig.xsd">
<UOM UnitsOfMeasurement="hp1" Name="abc"/></IFW>
```

Updating Data with the LoadIfwConfig Utility

To update data from the Pipeline Manager database:

1. Go to *Pipeline_home/bin*.
2. Run **LoadIfwConfig** using one of the following commands:
 - **Non-interactive mode**
`LoadIfwConfig [-i] input_file [-p] output_file`
 - **Interactive mode**
`LoadIfwConfig [read input_file] [update] [commit] output_file`

Exporting Data from the Database with the LoadIfwConfig Utility

To export data in a table, you must specify the primary field of the object in the input XML file. The utility exports all dependent data. For example, if you specify to export rate plans, the utility exports rate plan versions, rate plan configurations, price models, impact categories, and time zones.

This sample shows the contents of an XML file for exporting data:

```
<?xml version="1.0" encoding="UTF-8"?>
<IFW xmlns="http://www.portal.com/tools/pricing"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.portal.com/tools/pricing IfwConfig.xsd">
<UOM UnitsOfMeasurement="hp1" Name="abc"/></IFW>
```

To export data from the Pipeline Manager database:

1. Go to *Pipeline_home/bin*.
2. Run **LoadIfwConfig** using one of the following commands:
 - **Non-interactive mode**
`LoadIfwConfig [-i] input_file [-f] [-r] [-o] output_file`
 - **Interactive mode**
`LoadIfwConfig [read input_file] [fetch] [write] [commit] output_file`

The exported objects are written in XML format.

Troubleshooting

Errors during data loading, deleting, and exporting are usually caused by the following:

- The XML data does not comply with the XSD structure.
- You are trying to load data into a table that does not exist.
- You are trying to load data but the prerequisite data does not exist. See ["About the Types of Objects to Migrate"](#).
- You are trying to delete data that has other data depending on it.

About Real-Time Rate Plans

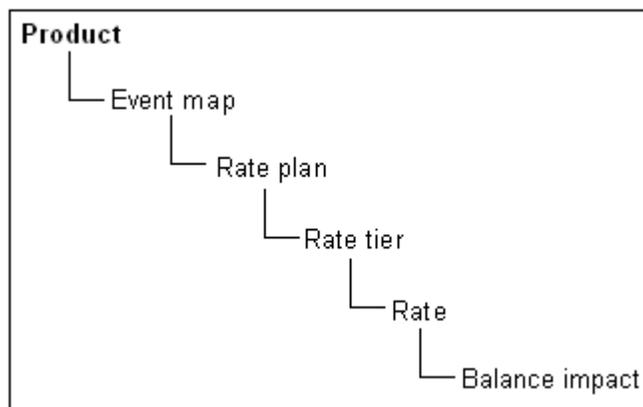
This chapter describes rate plans used in Oracle Communications Billing and Revenue Management (BRM) real-time rating.

Before reading this chapter, you should be familiar with the basic price list concepts. See "[About Creating a Price List](#)".

About Organizing Real-Time Rates in a Product

You organize rates in a product in a hierarchical structure as shown in [Figure 11-1](#):

Figure 11-1 Product Rate Structure



This structure enables you to create multiple components at each level; for example, you can rate multiple events in a single product, create multiple rates based on dates, and create multiple balance impacts for a rate based on previously rated quantities.

About the Event Map

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

- Monthly Cycle Forward Event
- IP Dialup Event

See "[Mapping Event Types to Services](#)".

About Real-Time Rate Plans

Usually, you use one rate plan per event. A rate plan includes the following:

- The rate or rates that apply to the event
- The currency to use for the event
By default, rate plans use the system currency.
- Tax information, such as when to calculate the tax, and the tax code
Assigning tax codes to rate plans enables you calculate taxes more accurately.
- For cycle forward fees, billing in advance
See "[Charging Cycle Forward Fees in Advance](#)".

You can use multiple rate plans to choose a rate plan based on event data (for example, telephone call origins and destinations). See "[Using Event Attributes to Rate Events in Real Time](#)".

In rare cases, you might need to use custom event analysis code to specify how to rate an event. See "[About Custom Event Analysis](#)".

You use rate plans to define how much to charge for an event. Rate plans include a currency, a tax code, and at least one rate tier. See "[About Organizing Real-Time Rates in a Product](#)".

Important:

- Rate plan names and descriptions (for example, the names of rate plans, rate tiers, and date ranges) can include a maximum of 255 characters.
 - If you use pipeline batch rating to rate events, you must use the same rate plan names in your real-time rate plans and in the pipeline rate plans.
-
-

Specifying the Rate Plan Currency

You specify the currency in which each customer's fees are charged. See "Managing system and account currencies" in *BRM Managing Customers*.

Specifying the Rate Plan Tax Code

You can assign a tax code to enable BRM or a third-party tax calculation package to calculate taxes. See "About Tax Codes" in *BRM Calculating Taxes*.

About Rate Tiers

You use rate tiers to define when and for how long rates are valid and which rates are applied first.

[Figure 11-2](#) shows a rate plan that includes three rate tiers. Each of the rate tiers is valid for a range of dates:

Figure 11–2 Rate Plan with Three Rate Tiers

The screenshot shows the 'Rate Plan Properties' dialog box. At the top, the currency is set to 'US Dollar [840]'. Below this, the 'Rate Structure' section lists three rate tiers: '9/1/02 - 12/31/02 Rate Tier', '1/1/03 - 1/31/03 Rate Tier', and '2/1/03 - 12/31/03 Rate Tier'. The 'Plan Detail' section includes a 'Name' field with 'IP Fax Rate Plan', a 'Taxes' section with 'Tax when: No Taxes' and an empty 'Tax code' field, and an 'In Advance Billing' section with 'Don't bill in advance' selected and 'Charge cycle fees' set to an empty field 'Days' in advance of billing cycle. Buttons for 'Add...', 'Delete', 'Change...', 'Move Up', and 'Move Down' are visible on the right side of the Rate Structure list. At the bottom are 'OK', 'Cancel', and 'Help' buttons.

For each rate tier, you can specify one of the following:

- **Absolute date range.** This rate tier is in effect only in the dates specified (for example, March through September).
- **Relative date range.** This rate tier is in effect only in a range of days relative to the product purchase (for example, for 30 days after the purchase date).

Note: For either absolute or relative date ranges, you can specify that a rate tier's valid period starts immediately and never ends. In effect, this makes the rate tier always valid.

You need more than one rate tier only in the following cases:

- To apply a balance impact for different resources in a specific order. For example, if you give customers free hours, you can prioritize rate tiers to charge for hours before currency.

- If you create rates based on date or time. For more information about rate tiers, see ["Real-Time Rating Based on Date and Time"](#).

About Real-Time Rates and Balance Impacts

A rate defines the balance impact that results when the event is rated (for example, \$2 for a one-hour dialup session). For information about balance impacts, see ["Specifying How an Event Affects an Account Balance"](#).

You usually use only one balance impact per rate, but you can specify multiple balance impacts for a single rate. For example:

- \$1 per fax
- 10 points toward free faxes

In addition, you can group balance impacts in terms of quantity. For example:

- For the first 10 faxes:
 - \$1 per fax
 - 10 points toward free faxes
- For the next 90 faxes:
 - 50 cents per fax
 - 25 points toward free faxes
- For 100 or more faxes:
 - 5 cents per fax
 - 50 points toward free faxes

For more information, see ["Real-Time Rating Based on Event or Resource Quantity"](#).

Specifying How an Event Affects an Account Balance

You specify the impact each event has on an account balance. You define each balance impact in terms of resources, general ledger IDs, proration, sponsorship, and discounts. In addition, you can assign an impact category to each balance impact.

The balance impact is the affect that rating has on an account balance. A balance impact can be positive or negative:

- A positive, or *credit*, impact is a charge to the customer (for example, \$2 for an hour of Internet usage). The customer owes you the amount of the credit balance impact.
- A negative, or *debit*, impact is a resource given to the customer (for example, 10 free Internet access hours per month). You owe the customer the amount of the debit balance impact.

About Scaled Impacts

An event that uses a *scaled* resource amount impacts an account balance immediately when the event starts and accumulates until the event ends. For example, BRM starts rating an IP telephony call as soon as it starts. This enables BRM to track the impact to the account as the event is rated.

A scaled amount affects a resource balance by multiplying the scaled amount value by the event quantity. For example:

Scaled rate of \$1 per hour X event quantity of 10 hours = balance impact of \$10

Use scaled impact when a resource is calculated per month, per hour, or by customer usage:

- \$1 per faxed page
- \$19.95 per month for interactive games service
- 10 free hours per month for Internet service
- \$.50 per email sent by the customer
- Purchase rates for products that customers can purchase in quantity

For example, you might sell a product that provides 100 hours of service a month. If customers can purchase two of these products for a total of 200 hours, use the scaled impact for the purchase rate.

As a rule, when you do not know the exact amount that will be rated, specify scaled impact.

Even though monthly rates are consistent from one month to the next, you still use a scaled impact. This is because they charge an amount per event; in this case, a monthly event generated by BRM:

Scaled rate of \$10 per month X event quantity of 1 month = balance impact of \$10

About Fixed Balance Impacts

A *fixed amount* is a predefined amount that is always applied to the account when any amount of the event quantity is rated.

Use a fixed impact to define rates based on occurrence. These are amounts that do not change with usage or with the quantity of events being rated. For example:

- \$5 installation fee
- \$10 cancellation fee
- 10 hours of free service when you register for a service

An event that uses a fixed amount does not impact the account balance until the event ends. This enables BRM to calculate variable fees for the same event type. For example, when a single balance impact has both a fixed amount and a scaled amount, BRM can sum the impact before the account balance is affected.

Specifying Multiple Balance Impacts for a Single Rate

You can specify more than one balance impact per rate:

- For the same event, you can apply a balance impact to a currency resource and to a non-currency resource. For example, you might create a product that charges \$20 per month and gives the customer 10 free Internet access hours. In that case, the same rate would have two balance impacts: one for US dollars and one for Internet hours.
- For the same event, you can apply a balance impact based on quantity. For example:
 - For 0 to 10 faxes, charge \$0.50 per fax.
 - For 11 or more faxes, charge \$0.10 per fax.

For more information, see "[Real-Time Rating Based on Event or Resource Quantity](#)".

Specifying the Validity Period of Granted Resources

You specify validity periods for balance impacts that grant resources such as free minutes. The validity period defines when a granted resource is available for consumption by the subscriber's usage. The subscriber can consume the resource balance during the valid period only.

You can set the balance impact to start immediately, relative to when the resource is granted, or on first usage (when the subscriber consumes the resource for the first time). Setting resources to start on first usage is useful, for example, to provide free resources for a limited time, beginning when the subscriber starts consuming the resource. For more information, see "[About Balance Impacts That Become Valid on First Usage](#)".

Assigning Impact Categories to Balance Impacts

You use impact categories to apply different balance impacts when using the same rate plan. For example, to rate calls made to two different countries, you create impact categories for each country. When the call event occurs, the impact category determines which balance impacts to apply.

If you are using a rate plan selector, you assign impact categories to balance impacts when you create a rate. When the rate plan selector selects a rate plan and impact category, only those balance impacts with the matching impact category are applied. See "[Using Event Attributes to Define Impact Categories](#)".

Assigning General Ledger (G/L) IDs to Balance Impacts

To collect information for your general ledger accounting system, you associate types of balance impacts with general ledger (G/L) IDs.

For example, when you create a usage rate, you can associate it with a G/L ID that records the revenue from all usage-rate balance impacts.

Allowing Charges to Be Sponsored

When charges are sponsored by a sponsor group, the sponsor group owner account receives the sponsored balance impacts generated by the group's member accounts. Therefore, you can use sponsor groups to let one customer pay part or all of other customers' service charges.

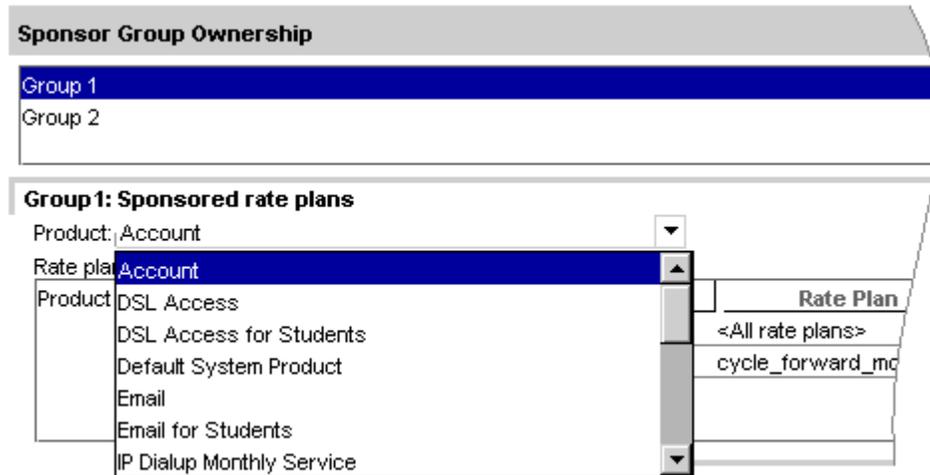
You define which accounts are sponsor group owners and which are members by creating sponsor groups in Customer Center. When you create a sponsor group, you select rate plans for the group to sponsor. You can select rate plans from all the products in your company's price list except the default product.

Tip: Creating sponsor groups is easier if all resources in the rates in sponsored rate plans are sponsorable. You might want to name your products so they can be identified as products that contain sponsorable rate plans.

[Figure 11-3](#) shows the **Product** list on Customer Center's **Sponsorship** tab. After selecting a product from this list, you select one or more of the product's rate plans for the group to sponsor.

Note: If your company supports branded service management, this list displays all the products in your company’s price list that are available for the sponsor group owner’s brand. If your company does not support brands, this list displays all the products in the price list.

Figure 11–3 Product List in Sponsorship Tab



One Sponsor per Product

Each products in a member account should have only one sponsor. If group A sponsors purchase charges for a service and group B sponsors monthly subscription charges for the same service, the purchase and cycle rate plans should be in different products. If both rate plans are in the same product, BRM assigns all the product’s charges to only *one* of the two sponsor groups. If the charges are assigned to group A, only the purchase fees for the service are sponsored. Members of group B must pay their own subscription fees.

For example, in [Figure 11–4](#), Product 2a rate plans are split between Dad’s group and Mom’s group. If BRM assigns Product 2a charges to Dad’s group, Kid’s account must pay its own wireless voice purchase fees even though Mom’s group is set up to sponsor them.

Figure 11–4 Sponsor Group Membership

Kid Sponsor Member: Account 0.0.0.1-13758 Status: Active

Summary Balance Payment Product Service Hierarchy Sponsorship

[Sponsorship](#) Sponsor group membership

Member of: 2 sponsor group(s)

Owner	Sponsor Group	Product	Rate Plan
Dad Sponsor Owner	Dad	Product 2a - Wireless Voice ...	cycle_forward_monthly
Mom Sponsor Owner	Mom	Product 2a - Wireless Voice ...	purchase

For more information about sponsorship, see "About sponsor groups" in *BRM Managing Accounts Receivable*.

You can also provide commissions by using service provider management. See ["About Remittance"](#).

About Balance Impacts That Become Valid on First Usage

You can set the validity period of resources granted by products and discounts to start on first usage: when subscribers begin consuming the resource balances for the first time.

For example, a deal grants 300 free minutes and 5 days of free video streaming. You set the free minutes to be valid immediately and set the streaming video to be valid when first used. A subscriber purchases the deal on June 1 and uses the video streaming service for the first time on June 5. The free minutes are valid starting June 1 and the streaming video is valid from June 5 to June 10.

You can also set the products and discounts that grant resources to start on first usage. There is no dependency between a product or discount's first usage start time and the first usage start time of the resource it grants. For information about products and discounts that start on first usage, see ["About Effective Periods That Start on First Usage"](#).

Resources whose validity period starts on first usage are added to the account balance at the time of the grant, but the validity period is not set until one of the following occurs:

- (Default) The first usage session in which the resource is consumed ends. The validity start time is set to the end time of the session.
- The resource is first authorized to be consumed. The validity start time is set to the start time of the first session.

The end time is set based on the end time that you configure for the resource and is set relative to the start time.

For more information, see ["About Setting Resource Validity Periods Based on First Usage."](#)

Note: If the product granting the resource also starts on first usage, when the product is first used and its validity period is set, resources that are granted by that product cannot be consumed by the first usage event during real-time rating. This is because real-time rating does not generate the purchase and cycle events that grant resources until after the first usage event is rated. However, resources granted by products are available for consumption during real-time discounting because discounting occurs after the purchase and cycle events have been processed. For information about products and discounts that start on first usage, see ["About Effective Periods That Start on First Usage"](#).

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

About Setting Resource Validity Periods Based on First Usage

Note: This section does not apply to products and discounts whose validity periods start on first usage. For information about them, see ["About Effective Periods That Start on First Usage."](#)

To set a resource validity period based on first usage, BRM must first set the period's start time. The end time is then set relative to the start time.

A resource validity period's start time is set during one of the following phases of the first usage session:

- **Stop Accounting:** (Default) The period's start time is set to the end time of the first usage session. This is recommended for typical validity periods.
- **Authorization:** The period's start time is set to the start time of the first usage session. This is recommended for short validity periods, such as one-day promotions. It prevents users from extending the promotional period by leaving the usage session open and thus preventing BRM from setting the validity period's end time.

Caution: If usage does not occur after the validity period's start time is set during the authorization phase (for example, a phone call is authorized but not answered), the validity period's start time nonetheless remains in effect and cannot be reset.

To specify when the start time is set, use the **SetFirstUsageInSession** business parameter. See ["Setting Validity Start Time to the Start Time of the First Usage Session."](#)

Setting Validity Start Time to the Start Time of the First Usage Session

By default, start times for resource validity periods based on first usage are set to the end time of the first usage session. Alternatively, they can be set to the start time of the session. You change the time they are set to by using the **pin_bus_params** utility. For information about that utility, see "pin_bus_params" in *BRM Developer's Guide*.

To set validity start time to the start time of the first usage session:

1. Go to the `BRM_Home/sys/data/config` directory, where `BRM_Home` is the directory in which BRM is installed.
2. Run the following command, which creates an editable XML file from the **activity** instance of the `/config/business_params` object:

```
pin_bus_params -r BusParamsActivity bus_params_act.xml
```

This command creates the `bus_params_act.xml.out` file in your working directory. To place this file in a different directory, include the path in the file name.

3. Open the `bus_params_act.xml.out` file.
4. Search for the following line:


```
<SetFirstUsageInSession>disabled</SetFirstUsageInSession>
```
5. Change **disabled** to **enabled**.
6. Save the file as `bus_params_act.xml`.

7. Go to the *BRM_Home/sys/data/config* directory, which includes support files used by the **pin_bus_params** utility.
8. Run the following command, which loads this change into the */config/business_params* object:

```
pin_bus_params PathToWorkingDirectory/bus_params_act.xml
```

where *PathToWorkingDirectory* is the directory in which **bus_params_act.xml** resides.

Caution: BRM uses the XML in this file to overwrite the existing **act** instance of the */config/business_params* object. If you delete or modify any other parameters in the file, those changes affect the associated aspects of the BRM activity configuration.

9. Read the object with the **testnap** utility or Object Browser to verify that all fields are correct.

See "Using testnap" in *BRM Developer's Guide* for general instructions about using the **testnap** utility. See "Reading Objects by Using Object Browser" in *BRM Developer's Guide* for information about how to use Object Browser.
10. Stop and restart the Connection Manager (CM). For more information, see "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.
11. (Multischema systems only) Run the **pin_multidb** script with the **-R CONFIG** parameter. For more information, see "pin_multidb" in *BRM System Administrator's Guide*.

About First-Usage Start Time for Shared Resources

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

About Synchronizing First-Usage Validity of Resources in Deals

You can synchronize the validity periods of all resources granted in a deal that have first-usage start times. This sets the validity of all granted resources with first-usage start times when one of those granted resources is first consumed.

For example, a deal grants free minutes and free text messages and both are set to start on first usage. A subscriber purchases the deal and on June 5 uses the free minutes by making a phone call. When the call is rated, the validity periods for both free minutes and text messages are set to start on June 5.

Only first-usage resources in the deal that is used to rate the event are synchronized. The validity periods of other resources and first-usage resources granted by other deals are not changed.

If a resource with a first-usage start time is granted after the validity period of other first-usage resources has already been set, the validity of the newly granted resource is synchronized with the existing validity periods.

To synchronize first-usage resources, you select **Align deal resource validity on first usage** in the **Deal Attributes** tab of Pricing Center.

About Restricting Resource Validity End Time to the Product or Discount End Time

For resources that start on first usage, you can restrict the resource validity end times so they do not extend beyond the validity end time of the products or discounts that grant the resources. This ensures that the resource balance cannot continue to be consumed after the product or discount expires.

Note: When a product or discount is *canceled* , the validity period end time of resources granted by that product or discount is set to the time of the cancellation.

For more information, see ["About Restricting the End Time of Granted Resources That Start on First Usage"](#).

About Fold Events

A fold event is a special internal event that usually occurs at the end of the accounting cycle. A fold event can be configured to either change a resource balance (currency or non-currency) to zero or convert the resource balance into other resources.

For example, you can create a fold event to cancel unused free hours at the end of each month. You can also create a fold event to credit frequent flyer miles for every \$500 a customer spends on a service.

Fold events are triggered at the following times:

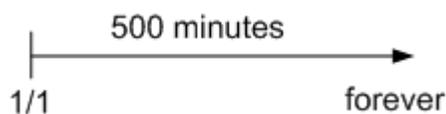
- During regular billing
- At product cancellation
- When Bill Now is invoked

When a fold event occurs, BRM checks whether the resource sub-balance or bucket is valid or expired. The resource is folded if the sub-balance is expired or is always valid.

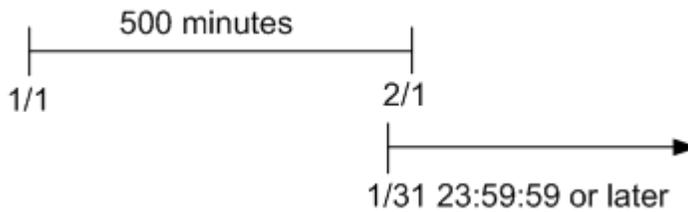
For example,

- 500 free minutes are valid forever; therefore the resource can be folded at any time as shown in [Figure 11-5](#).

Figure 11-5 Free Minutes Valid Forever



- 500 free 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 11-6](#).

Figure 11–6 500 Free Minutes Expiring on February 1st

Important: If you use delayed billing, folds occur at the end of the delayed billing period. This is to ensure that delayed events can continue to consume granted resources before final billing is run. However, free resources are granted at the beginning of each cycle, before the fold event occurs. If the newly granted resources do not have a configured validity period (that is, their validity period never ends), the fold event will remove those resources.

When you use fold events to remove unused resources and you also use delayed billing, you should always configure a validity period for the granted resources. This prevents fold events from removing resources that are granted at the beginning of a cycle before final billing is run.

You can configure fold events to convert resource R1 to R2. You can also convert R2 to R3 if resource R2 is available or tracked in the account's balance group *before* the fold event is triggered.

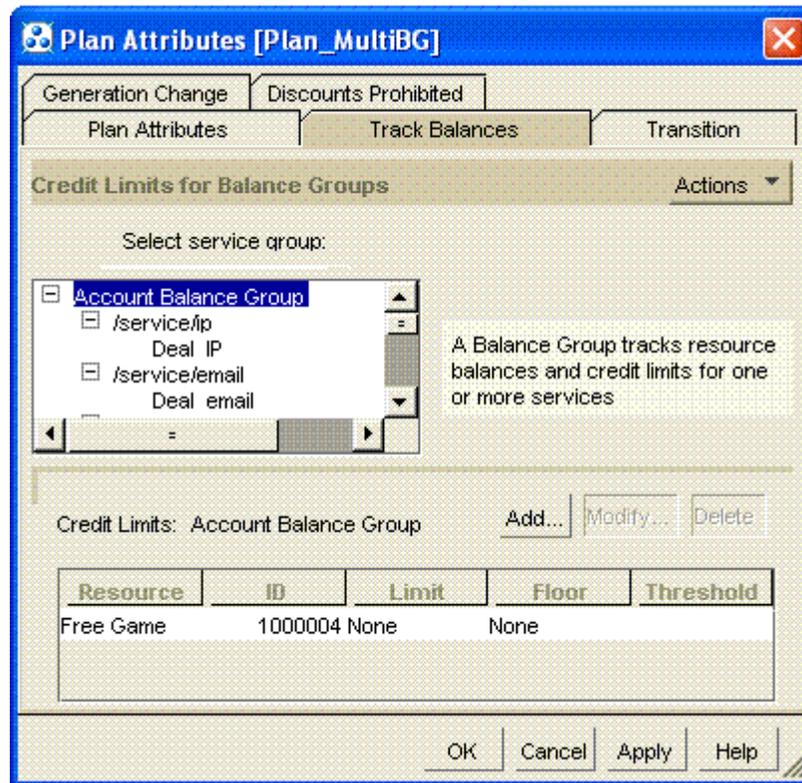
For example, a price plan has a cycle forward event that charges a \$100 cycle fee and grants 1000 free minutes. When you create an account with this plan, the account's balance group contains two resources: USD and Free Minutes. If you configure a fold event to:

1. Convert the free minutes (R1) to free games (R2)
2. Convert the free games (R2) to Euro (R3)
3. Convert the free minutes (R1) balance to zero

The free games resource (R2) is not converted to R3 in step 2 because it is not available in the account's balance group when the fold is triggered, even though Free Games is added to the account's balance group in step 1.

To fold free games to Euro, you add Free Games to the plan in the **Track Balances** tab in the Plan Attributes window as shown in [Figure 11–7](#). When the account is created, Free Games is added to the account's balance group and its balance is set to zero (0).

Figure 11-7 Plan Attributes Track Balances Tab



By default, an account's resource balances are folded in ascending order based on the resource ID. You can change this order by modifying the policy opcode, `PCM_OP_SUBSCRIPTION_POL_SPEC_FOLD`. See ["Customizing the Order to Apply Folds"](#).

You can create a fold to charge for overuse of free resources before the customer cancels a product.

For more information about applying folds, see ["About Applying Folds"](#).

Ensuring That All of an Event Is Rated

BRM cannot apply a rate if the credit limit for its resource has been reached. However, there might be times when an event, or part of an event, has not been rated, and all credit limits have been reached. To charge for the unrated event or event portion, you create a *limit override rate*. A limit override rate is used even though a credit limit has been reached.

In effect, a limit override rate sets the baseline charge for an event. When no other rates apply because of credit limits, you charge a default amount for the event.

Real-Time Rating Based on Multiple RUMs

This chapter shows how to use Oracle Communications Billing and Revenue Management (BRM) to set up products to rate events based on multiple ratable usage metrics (RUMs).

Before you read this chapter you should be familiar with rating and price lists. See ["About Creating a Price List"](#).

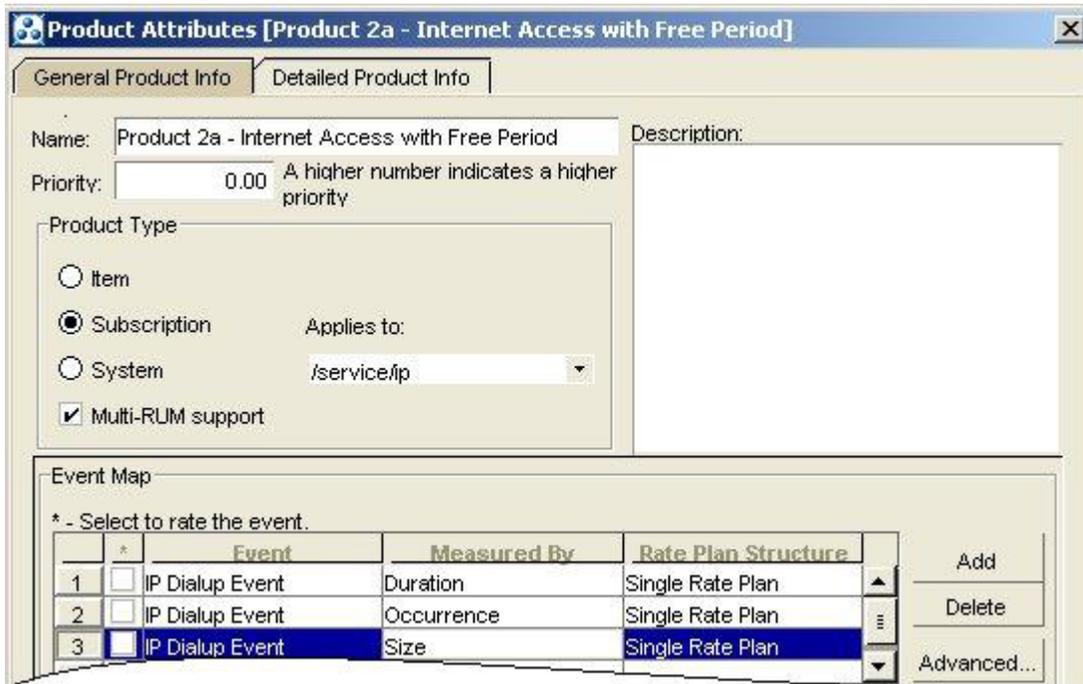
About Rating Based on Multiple RUMs

By default, BRM rates an event by using a single RUM that is specified in the product's usage map. You can also set up your products to rate an event based on multiple RUMs. For example, you can set up a Session Event with both Occurrence and Duration selected as RUMs by which the event is measured.

To rate an event by using multiple RUMs, you map the event to the RUMs in the product's usage map.

[Figure 12-1](#) shows the **Multi-RUM support** option selected and multiple IP Dialup events with different RUMs defined:

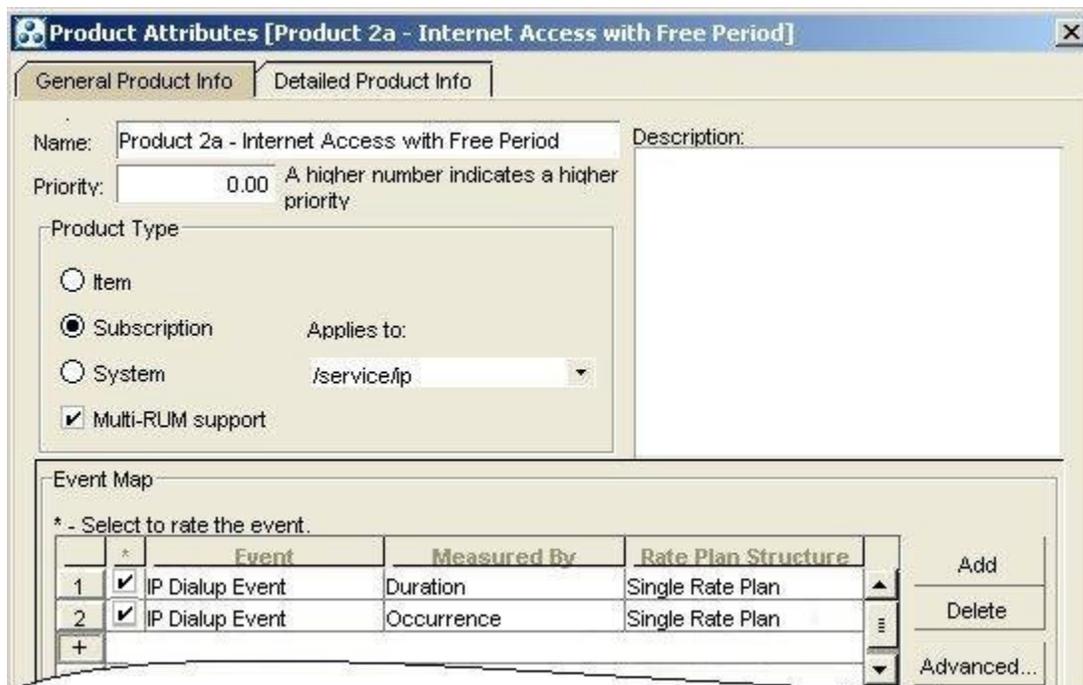
Figure 12-1 Multi-RUM Support for IP Dialup Event



To apply the RUMs, you map the event to the RUMs and the rate plan in the product’s usage map.

Figure 12-2 shows a usage map for an Internet Access product with two IP Dialup events mapped to **Duration** and **Occurrence**, with both events selected for rating:

Figure 12-2 Rating IP Dialup Events Using Duration and Occurrence



For each event to be rated, click the check box in the * column, located between the event number column and the **Event** column.

Important: Specifying multiple RUMs for the same event type and selecting events to be rated are two separate functions, each handled in the Event Map.

When an event is rated using multiple RUMs, a balance impact is applied for each RUM per each resource. For example, if two resources are impacted, two separate balance impacts are created.

For information about setting up products with multiple RUMs, see "[Applying Multiple RUMs to Rate an Event](#)".

Applying Multiple RUMs to Rate an Event

To rate an event by using multiple RUMs, you map the event to the RUMs in the product's usage map.

To use multiple RUMs to rate an event, you do the following:

1. Define the RUMs available for rating. See "[About Ratable Usage Metrics](#)".
2. Map the event to the RUMs in the product's usage map. See "[Example of Mapping an Event to Multiple RUMs](#)".

Example of Mapping an Event to Multiple RUMs

When you create a product, you specify the events to rate for the service, and the rate plans that determine how to charge for those events. To rate an event by using multiple RUMs, you associate the event with the RUMs and the rate plans.

You can map the same event to multiple RUMs by using different rate plans. For example, you can specify a rate plan to charge for a fax event based on the number of pages faxed and another rate plan to charge based on the duration of the fax session.

To map an event to RUMs:

1. Specify the RUMs in the product's usage map.

In [Figure 12-3](#), two IP Dialup events are mapped to **Duration** and **Occurrence** with different rate plans, with both events selected for rating:

Figure 12-3 Multiple RUM Event Mapping Example



2. Indicate the events to be rated by clicking the check box in the * column, located between the event number column and the event type column.
3. When you finish creating products, deals, and plans, commit your price list to the BRM database.

Rating Based on Multiple RUMs with Pipeline Manager

This chapter shows how to use Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager to set up products to rate batch events based on multiple ratable usage metrics (RUMs).

Before you read this chapter, you should be familiar with rating and price lists. See ["About Creating a Price List"](#).

About Rating Based on Multiple RUMs with Pipeline Manager

To rate a real-time event by using multiple RUMs, you map the event to the RUMs in the product's usage map. However, to rate an event using multiple RUMs with Pipeline Manager, you set up multiple Price Model steps. In other words, you configure pipeline rating within the individual price models.

For each RUM, the data is set in the Event Data Record (EDR) before it is sent to the batch pipeline. Pipeline Manager then uses this data to do the rating based on each individual RUM. Each RUM is fully rated before rating the next RUM.

For more information about applying multiple RUMs to rate a batch event, see ["Applying Multiple RUMs to Rate a Batch Event"](#).

Applying Multiple RUMs to Rate a Batch Event

To rate a batch event by using multiple RUMs, you set up multiple Price Model steps.

To use multiple RUMs to rate a batch event, do the following:

1. Define the RUMs available for rating. See ["About Ratable Usage Metrics"](#).
2. Set up price model steps. The following example shows a multi-RUM rate plan model created for a standard GPRS delayed event:

- Duration RUM: 0.02 EUR/min
- Volume Sending RUM: 2 EUR/MB
- Volume Receiving RUM: 2 EUR/MB
- In this case, you set up the following price model steps:

Configure the pricing for the **Duration RUM** as follows:

Beat: 60

Charge: 0.02

Charge Base: 60

Configure the pricing for the **Volume Receiving RUM** as follows:

Beat: 2048

Charge: 2.00

Charge Base: 1048576

Configure the pricing for the **Volume Sending RUM** as follows:

Beat: 2048

Charge: 2.00

Charge Base: 1048576

Real-Time Rating Based on Date and Time

This chapter explains how to use Oracle Communications Billing and Revenue Management (BRM) to set up rates based on date and time.

Before you read this chapter, you should be familiar with rating and price lists. See ["About Creating a Price List"](#).

About Rating Based on Date and Time

BRM rates billable events by capturing data about the events (for more information, see ["About Creating a Price List"](#)), such as the date and time each event occurred. Because BRM is aware of the date and time of an event, the rating opcodes can use the date and time to determine how to rate the event.

Ways to Specify When Rates Are Valid

You can specify when rates are valid in the following ways:

- **In products.** You can specify when a product is valid, including all the rates it contains. See ["Restricting When a Product Is Available"](#).
- **In rate tiers.** You can use rate tiers to specify a set of rates, each based on a specific time or day. See ["Using Rate Tiers to Set Up Rating Based on Date and Time"](#).

When you create rates based on date and time, you can specify how to recognize the time that the event occurred (for example, at the start, end, or duration of the event).

You specify a time-of-day mode to indicate when a rate should be applied relative to either an event's start time, its end time, or the difference between the two. This is especially useful when an event overlaps a time boundary where a rate changes.

For example, 12 a.m. Monday might be the boundary between a special weekend rate and a regular weekday rate:

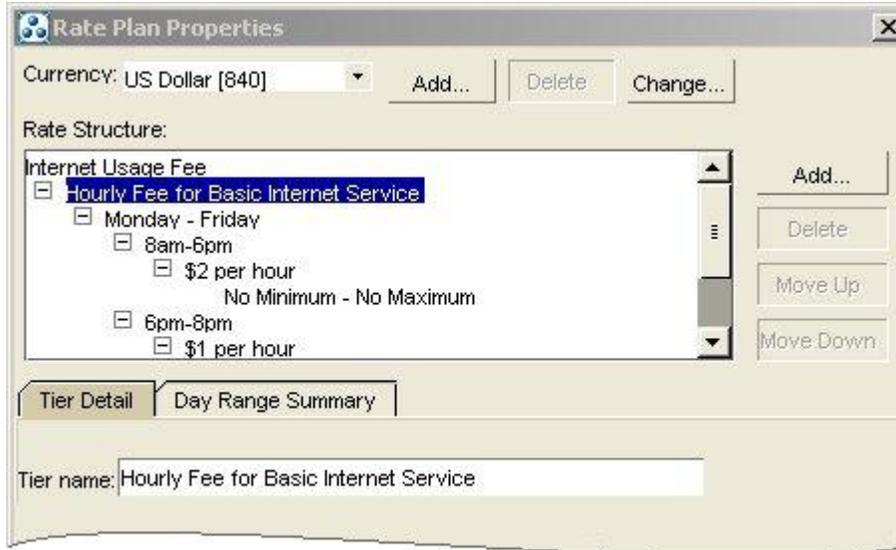
- To apply the special weekend rate to the entire event, specify **Start Time**.
- To apply the regular weekday rate to the entire event, specify **End Time**.
- To apply the weekend rate to the portion of the event before midnight and the weekday rate after midnight, specify **Timed**.

Note: Midnight (12 a.m.) is shown as 00.00.00.

Using Rate Tiers to Set Up Rating Based on Date and Time

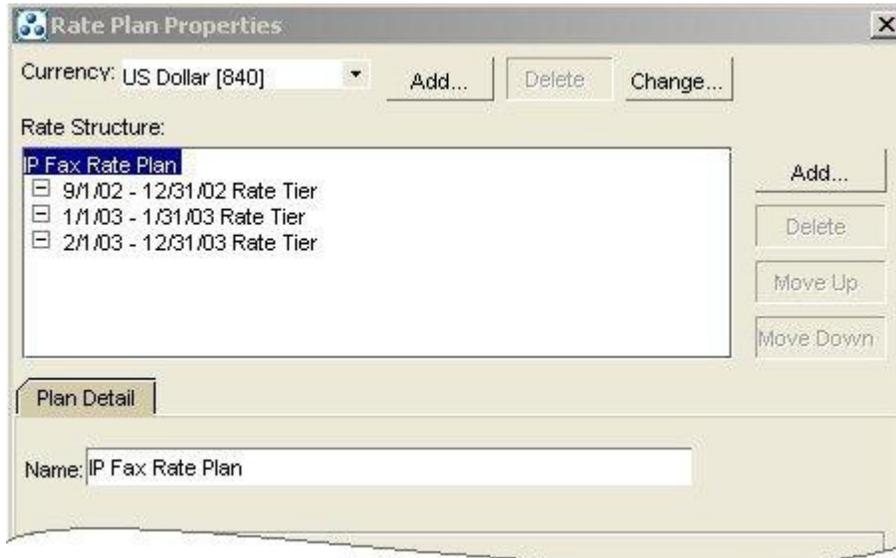
You use rate tiers to set up valid dates and times for rates. [Figure 14-1](#) shows a rate tier that includes two different rates:

Figure 14-1 Rate Plan with Two Different Rates



You can include multiple rate tiers in a rate plan. [Figure 14-2](#) shows a rate plan with three rate tiers, each valid for a different set of dates:

Figure 14-2 Rate Plan with Three Rates



Specifying Which Rate Tiers Are Applied First

As you create rate tiers, they appear on a list. BRM applies rates from the top of the list down. You can change the order in which rate tiers are applied by changing their places in the list shown in [Figure 14-2](#).

For example, at the top of the rate tier list you could insert a promotional rate tier valid only until the end of May 2003. BRM applies that rate only until May 31, 2003. After that, the promotional rate tier is no longer valid, and BRM no longer applies it.

Specifying How Long Rate Tiers Are Valid

You specify how long a rate tier is valid by specifying a duration. You can specify that a rate tier is always valid or is valid for either an absolute or a relative date range. [Figure 14-2](#) shows three rates with validity specified by date.

For example, your basic IP Fax rate tier should not change and always applies. You could specify an absolute date range of 12/1/01 through 12/31/01 for a promotional holiday rate tier. You could also specify a relative date range of 30 days from purchase until 60 days from purchase to give your customers special rates during the second month.

Specifying When Rate Tiers Are Valid

You specify when a rate tier is valid by selecting ranges of dates, days of the week, and times of day. For example:

- For each rate tier, you specify a valid date range. For example, you can specify that a rate tier is valid for the entire year 2001.
- For each date range, you specify one or more valid day-of-the-week ranges. For example, you can specify that a rate tier is valid Monday through Friday.
- For each day-of-the-week range, you specify one or more valid time-of-day ranges. For example, you can specify that a rate tier is valid between 6 a.m. and 8 p.m.

Using Date, Day, and Time Ranges Together

Date ranges consist of a start date and an end date between which a rate tier is valid. For example, a promotional rate tier for usage fees is valid only for the month of January.

Day ranges consist of a list of days during which a rate is valid. For example, a rate tier for off-peak IP usage fees is valid only on weekends.

Time-of-day ranges consist of a start time and an end time between which a rate tier is valid. For example, an after-hours rate tier for IP usage fees is valid only between 10 p.m. and 6 a.m.

Within each date range, you can use multiple day ranges; within each day range, you can use multiple time-of-day ranges.

The following examples show how you can use date, day, and time-of-day ranges together. [Figure 14-3](#) shows some time ranges that specify free IP usage for the first month after an account is created. [Figure 14-4](#) shows some time ranges that specify IP usage rates effective after the first month.

Figure 14-3 Free IP Usage for First Month

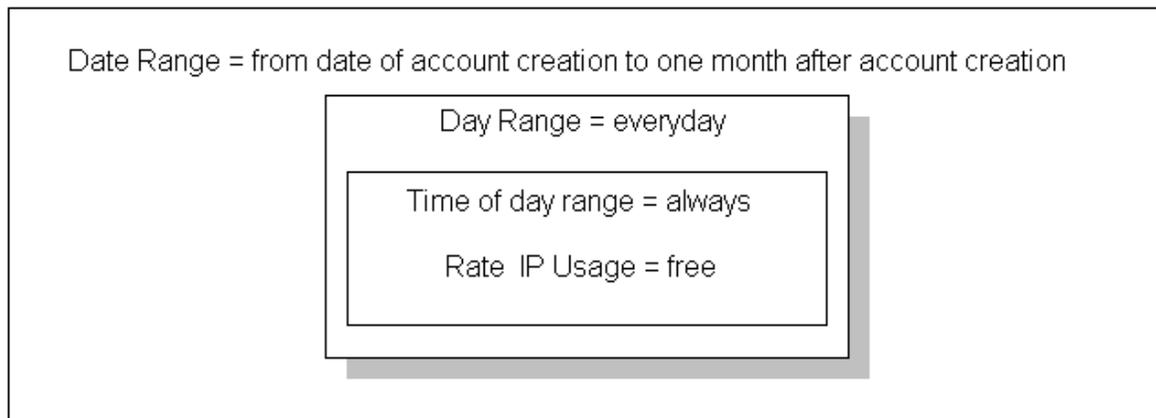
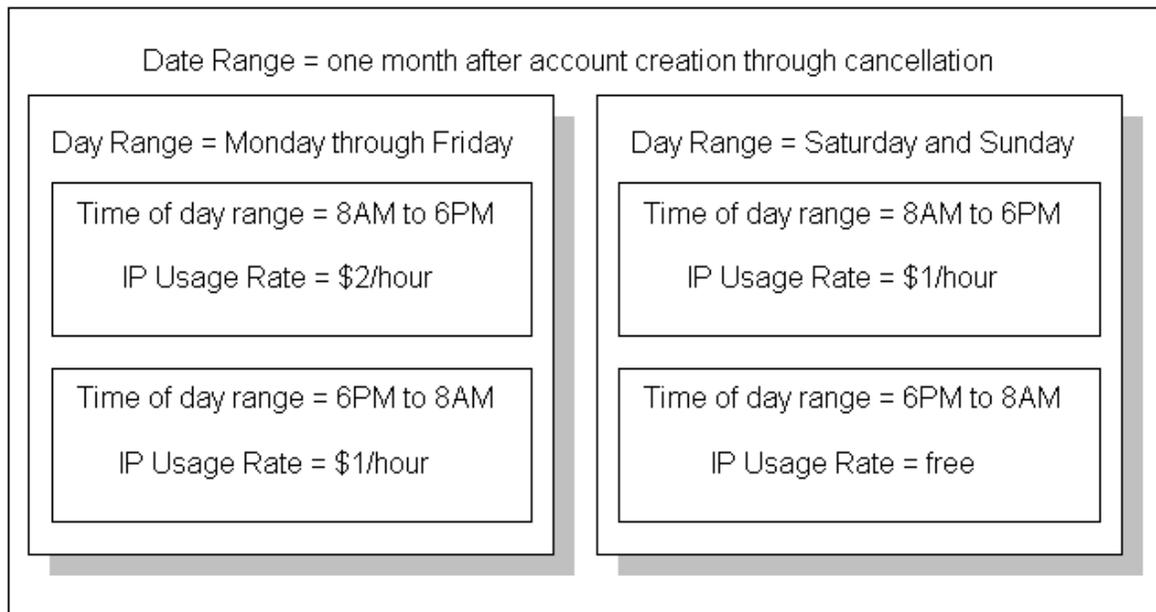


Figure 14-4 Time Range Based IP Usage Rates after First Month



Using the Correct Time Zone

Because BRM uses the date and time of an event to determine how to rate the event, using the correct time zone during rating is important. You specify whether real-time rating uses the *event* time zone or the *server* time zone to rate the event. By using time zone data, BRM can correctly calculate event start and end times before rating occurs.

Specifying a Time Zone Mode

When you create a product in Pricing Center, you specify which of the following time zone modes BRM uses for rating events whose rates are based on time of day:

- **Event:** BRM uses the time zone in which the event occurs. For example, if a user in California logs on to a terminal server located in California, BRM uses California time, even if the BRM server that rates the event is in New York. This is the default time zone mode.
- **Server:** BRM uses the time zone in which the BRM server that rates the event is located. For example, if a California user logs on to a terminal server located in California but the BRM server that rates the event is in New York, BRM uses New York time. For information about setting your system's server time zone, see "[Resetting the Server Time Zone](#)".

For information about setting the time zone mode, see the Pricing Center Help.

Resetting the Server Time Zone

The server time zone is where the server you use for rating resides.

By default, the server time zone is set to **America/Los_Angeles**. If your rating server is not in that time zone, reset the server time zone to the time zone ID indicating the server location.

Important: BRM supports only the time zone IDs defined in instances of the Java `TimeZone` class.

To reset the server time zone:

1. Get a list of time zone IDs in the Java `TimeZone` class by running the following function:

```
TimeZone.getAvailableIDs ()
```

2. In the list, find the ID for the time zone in which your server resides and write it down.
3. Open the `BRM_home/sys/cm/timezones.txt` file, where `BRM_home` is the directory in which BRM is installed.

This file lists all the time zone IDs available for use in your system.

4. If the ID for the time zone in which your server resides is *not* in the `timezones.txt` file, do the following:
 - a. Open the `BRM_home/sys/cm/sample_timezone.txt` file.
 - b. Add an entry to the `sample_timezone.txt` file for the time zone in which your server resides by following the instructions in the file.

Important: BRM supports only the time zone IDs defined in instances of the Java `TimeZone` class.

Each time zone ID entry in the `sample_timezone.txt` file contains 13 mandatory parameters. The value of the `tzname_name` parameter must match the name of a time zone ID in the Java `TimeZone` class. To get a list of time zone IDs in the `TimeZone` class, run the `TimeZone.getAvailableIDs()` function.

Note: To add multiple time zone IDs to the file, enter each time zone ID entry on a separate line.

- c. Save and close the `sample_timezone.txt` file.
5. Open the `BRM_home/sys/cm/pin.conf` file.
6. If you added an entry to the `sample_timezone.txt` file in step 4, add the following entry to the `pin.conf` file:

```
- fm_rate timezone_file sample_timezone.txt
```

7. Find the following entry in the `pin.conf` file:

```
-rating_timezone server_timezone_ID
```

where `server_timezone_ID` specifies the ID of the time zone in which the BRM server performing rating is located.

8. Replace the current `server_timezone_ID` value with the time zone ID of your BRM rating server.

For example, if your rating server is in Saskatchewan, use the following time zone ID:

```
- fm_rate rating_timezone Canada/Saskatchewan
```

Note: To add multiple time zones to the file, add separate `-fm_rate rating_timezone` entries.

9. Save and close the file.
10. Stop and restart the CM.

Configuring Applications to Generate Time Zone Data

When a BRM client application or service integration client (such as RADIUS Manager and Universal Event Loader) generates an event, that event includes time zone data. You must configure event-generating applications to generate the correct time zone data. This means creating an entry in a `pin.conf` or properties file. See the configuration section of your client application or the service integration component documentation for details.

Using Event Start Time to Determine a Product's Validity

By default, in real-time rating an event's end time is used to determine a product's validity.

You can specify that the event start time be used for determining the product's validity by adding an entry to the CM configuration file.

To use the event start time to determine the product's validity:

1. Open the *BRM_Home/sys/cm/pin.conf* file.
2. Add the following entry:
 - **fm_rate use_prods_based_on_start_t /event**

You can use **/event** to use the start time for all events, or use a subclass of **/event** to specify just events of a given type.

3. Stop and restart the CM.

Real-Time Rating Based on Event or Resource Quantity

This chapter describes how to set up quantity-based rating in Oracle Communications Billing and Revenue Management (BRM).

Before you read this chapter you should be familiar with rating and price lists. See ["About Creating a Price List"](#).

About Quantity-Based Rating

You use quantity-based rating when you must rate events based on previous rating or on the current balance of a resource. For example:

- \$1 per fax for the first 10 faxes.
- 50 cents per fax for the next 90 faxes, up to and including the 100th fax.
- Free faxes for 101 or more.

To set up quantity-based rating, you group together balance impacts into *quantity discount brackets*, which are applied for a range of event quantities. For each range, you apply a different bracket group of balance impacts and you charge customers differently for each event quantity.

Note: Quantity-based rating applies only to events rated in real time. You can use discounts to achieve many of the same results for both events rated in real time and events rated in batch by the pipeline. See "About discounts" in *BRM Configuring Pipeline Rating and Discounting*.

Selecting a Quantity Discount Bracket Based on Previous Rating

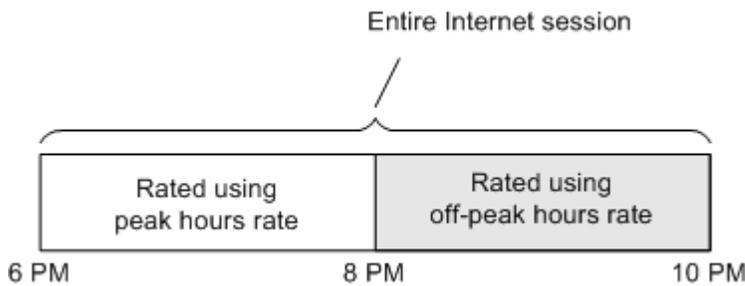
You can use the event quantity as the basis for selecting a quantity discount bracket by using the Continuous option or the Rate Dependent option. When BRM requires only one rate for an event, the Continuous or Rate Dependent options do the same thing. When BRM requires more than one rate for a given event, the difference between Continuous and Rate Dependent becomes important. For example, see the rates for Internet access shown in [Table 15-1](#):

Table 15–1 Connection Times and Discount Calculations

Peak Hours Rate: 6 a.m. - 8 p.m.	Off-Peak Hours Rate: 8 p.m. - 6 a.m.
Quantity Discount Bracket 1: No min-No max Balance Impact: \$1 per hour	Quantity Discount Bracket 1: 0-2 hours (first two hours) Balance Impact: \$0.50 per hour
Quantity Discount Bracket 1: No min-No max Balance Impact: \$1 per hour	Quantity Discount Bracket 2: 2+ hours (subsequent hours) Balance Impact: \$0.25 per hour

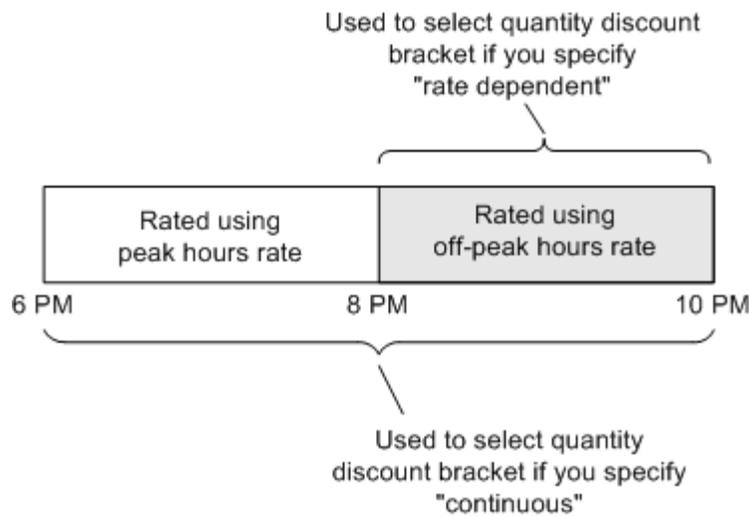
In the example shown in [Table 15–1](#), your customer connects to the Internet at 6 p.m. and disconnects at 10 p.m. The 2 hours between 6 p.m. and 8 p.m. are rated by the peak hours rate. The 2 hours between 8 p.m. and 10 p.m. are rated by the off-peak hours rate. [Figure 15–1](#) shows a timeline for this example.

Figure 15–1 Internet Session Spanning Peak and Off-Peak Rates



Which quantity discount bracket from the off-peak hours rate is used depends on whether you specify *Continuous* or *Rate Dependent* as shown in [Figure 15–2](#).

- Specify *Continuous* if the quantity used to select the quantity discount bracket for the off-peak hours rate will include *both* the quantity rate by using the peak hours rate and the quantity rated by using the off-peak hours rate.
- Specify *Rate Dependent* if the quantity used to select the quantity discount bracket for the off-peak hours rate will include *only* the quantity rated by using the off-peak hours rate.

Figure 15–2 Continuous Versus Rate Dependent Discounting

Note: The **Resource Balance** option isn't applicable because you aren't basing the charges on a resource balance in the customer's account.

Selecting a Quantity Discount Bracket Based on a Resource Balance

When you specify *Resource Balance*, BRM selects a quantity discount bracket based on the balance of a resource. For example, by using the following rates in [Table 15–2](#) to charge your customers for IP telephony:

Table 15–2 Rating Customers for IP telephony

Peak Hours Rate: 6 a.m. - 8 p.m.	Off-Peak Hours Rate: 8 p.m. - 6 a.m.
Quantity Discount Bracket 1: 0-10 hours of Internet time Balance Impact: \$0.10 per minute	Quantity Discount Bracket 1: 0-10 hours of Internet time Balance Impact: \$0.05 per
Quantity Discount Bracket 2: 10+ hours of Internet time Balance Impact: \$0.07 per minute	Quantity Discount Bracket 2: 10+ hours of Internet time Balance Impact: \$0.02 per hour

Between the hours of 6 a.m. and 8 p.m., the peak hours rate is used. If your customer has accumulated fewer than 10 hours of Internet time, your customer is charged \$0.10 per minute for IP phone calls. If your customer has accumulated more than 10 hours of Internet time, your customer is charged \$0.07 per minute for IP phone calls.

Between the hours of 8 p.m. and 6 a.m., the off-peak hours rate is used. If your customer has accumulated fewer than 10 hours of Internet time, your customer is charged \$0.05 per minute for IP phone calls. If your customer has accumulated more than 10 hours of Internet time, your customer is charged \$0.02 per minute for IP phone calls.

Note: Resource balance-based rating has limitations for cycle events. If a cycle event is canceled at the end of a quantity discount bracket, BRM bases the cancellation fees on the next quantity discount bracket. For example, if the pricing configuration includes different cycle fees based on the number of elapsed months, you can create a counter for the number of elapsed months: quantity discount bracket 1 for months 0 to 6 (the first six months), and quantity discount bracket 2 for months 6+. If a customer cancels during the sixth month, the cancellation fee is based on quantity discount bracket 2.

Example 1: Rating Based on the Quantity of an Event

To rate events based on their quantity or duration, use either the Continuous (total quantity) or Rate Dependent option. For example, an IP session event uses the following 3 quantity discount brackets to rate the duration of the session:

1. First 10 minutes = \$0.12 per minute
2. 10 - 20 minutes = \$0.07 per minute
3. Over 20 minutes = \$0.05 per minute

Figure 15-3 shows how to set up the brackets in Pricing Center:

Figure 15-3 Quantity Discount Brackets Configuration

The screenshot displays the 'Quantity Discount Brackets Configuration' window. At the top, the 'Rate Structure' is shown with a tree view containing 'cycle forward monthly', 'Monthly Service Fee', and 'Basic IP Session'. The 'Basic IP Session' item is selected and circled in green. Below this, three quantity discount brackets are listed: 'No Minimum - 10.00', '10.00 - 20.00', and '20.00 - No Maximum'. To the right of the tree view are buttons for 'Add...', 'Delete', 'Move Up', and 'Move Down'. Below the tree view, there are tabs for 'Rate Data' and 'Proration'. The 'Name' field contains 'Basic IP Session'. A checkbox labeled 'Overrides credit limit' is checked. At the bottom, the 'Quantity Discount Brackets' section shows 'Based on: Continuous' and 'US Dollar [840]'.

Notice that the maximum value in the first two brackets is the same as the minimum value in each subsequent bracket. Always set up balance impact groups this way to avoid errors in your price list.

With continuous rating, the maximum value does not represent an actual quantity; but rather a boundary between ranges that is not actually rated. In this example, the smallest fraction of time less than 10 minutes is rated in the first tier, and the smallest fraction of time greater than 10 minutes is rated by the second tier. Everything up to the maximum value in a range is included in that range, and anything over the maximum value is included in the next range.

The following examples show how to set up the values for each balance impact group.

Figure 15-4 shows the first discount bracket (0-10 minutes = \$0.12 per minute).

Figure 15-4 Quantity Discount Bracket 1

	Resource ID	GLID	Impact Category	S	P	D	Fixed Amount	Scaled Amount	Units
1	US Dollar [840]	Dialup Usage ...		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	0.00	0.12	Minute [2]

Figure 15-5 shows the second discount bracket (10-20 minutes = \$0.07 per minute).

Figure 15-5 Quantity Discount Bracket 2

	Resource ID	GLID	Impact Category	S	P	D	Fixed Amount	Scaled Amount	Units
1	US Dollar [840]	Dialup Usage ...		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	0.00	0.07	Minute [2]

Figure 15-6 shows the third discount bracket (20+ minutes = \$0.05 per minute).

Figure 15-6 Quantity Discount Bracket 3

	Resource ID	GLID	Impact Category	S	P	D	Fixed Amount	Scaled Amount	Units
1	US Dollar [840]	Dialup Usage ...		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	0.00	0.05	Minute [2]

Example 2: Rating Based on Total Event Quantities

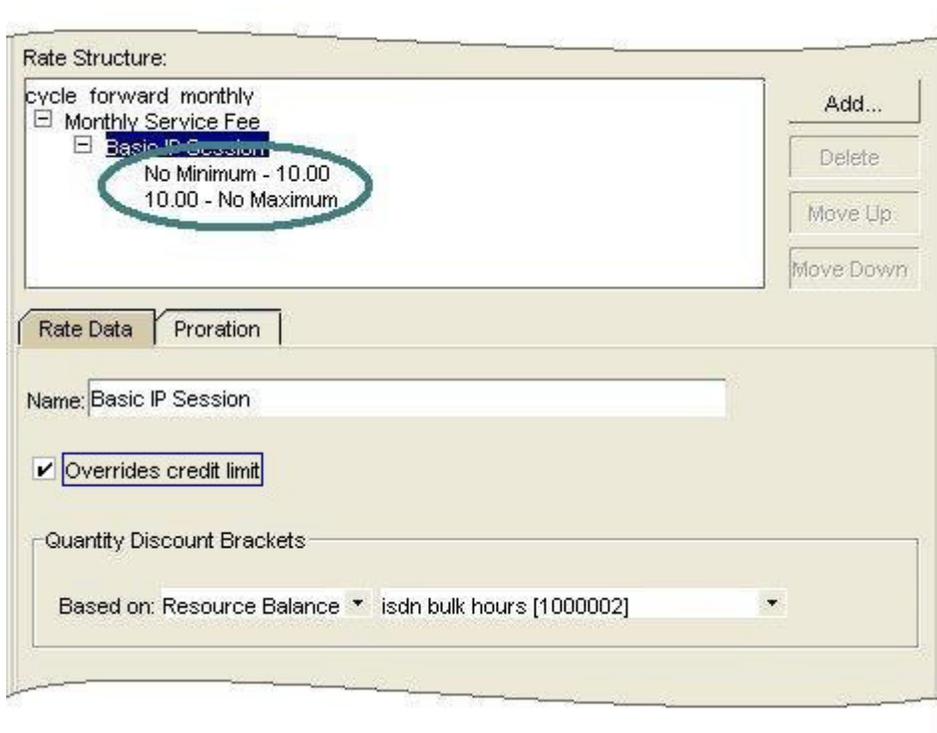
This example shows how to rate the total number of hours per month for an IP session by using two quantity discount brackets.

- Quantity Discount Bracket 1: 0-10 hours a month = \$0.10 a minute
- Quantity Discount Bracket 2: 10 + hours a month = \$0.05 a minute

Set up the quantity discount brackets based on the resource balance “ISDN bulk hours”. You base charges on the minutes per session and the monthly hourly total. Each bracket requires 2 balance impacts: 1 to track the charge per minute and 1 to track each hour of usage.

Figure 15-7 shows how to set up the brackets in Pricing Center:

Figure 15–7 Rate Structure Discount Bracket Configuration in Pricing Center



The following figures show how to set up individual balance impacts for each group:

Figure 15–8 shows the first discount bracket (0-10 hours = \$0.10 per minute).

Figure 15–8 Quantity Discount Bracket 1

	Resource ID	GLID	Impact Categ...	S	P	D	Fixed Amount	Scaled Amou...	Units
1	IP sessions [1000125]	undefin...		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1.00	0.00	None [0]

Figure 15–9 shows the second discount bracket (10+ hours = \$0.05 per minute).

Figure 15–9 Quantity Discount Bracket 2

	Resource ID	GLID	Impact Categ...	S	P	D	Fixed Amount	Scaled Amou...	Units
1	US Dollar [840]			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0.00	0.10	Minute...
2	IP sessions [1000125]			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1.00	0	None [0]

Important: Because you track the charges based on the monthly total, you also require an ISDN bulk hours fold event that removes the total hours balance each month.

Example 3: Rating Based on Total Number of Events

To rate events based on the total number of events that occur over a period of time, use the resource balance for that event. For example, if you charge customers for the total

number of IP sessions they have each month, create a custom resource "IP_Sessions" and use it to track the total number of events each month.

For example, an IP Session event is rated based on the number of sessions by using the following 2 quantity discount brackets:

- Quantity Discount Bracket 1: 0-10 sessions = no charge
- Quantity Discount Bracket 2: more than 10 sessions = \$0.10 per minute.

Figure 15-10 shows how to set up the brackets in Pricing Center:

Figure 15-10 Rate Structure Based on Number of Events

The screenshot displays the 'Rate Structure' configuration window. The 'Rate Structure' section is expanded to show a 'Monthly Service Fee' with two brackets: '0-10 free, 10 + \$0.10 minute' (highlighted in blue) and 'No Minimum - 10.00' (circled in green). The 'Rate Data' tab is selected, showing the name '0-10 free, 10 + \$0.10 minute', a checked 'Overrides credit limit' checkbox, and a 'Quantity Discount Brackets' section with 'Based on: Resource Balance' and 'IP sessions [1000125]'.

In the above example, sessions 1-10 are free. Each subsequent session is rated at \$0.10 per minute. When the 11th session begins, the resource "IP Sessions" is incremented and the second balance impact containing the scaled amount (\$0.10 per minute) is applied to the account balance. This way, when the next session occurs, the second rate is used.

Note: Scaled amounts are applied to the account as soon as the event occurs. Fixed amounts are applied to the account after the event occurs.

The following examples show how to set up the balance impacts for each bracket.

Figure 15-11 shows the first discount bracket.

Set up the first bracket "No Minimum - 10.00" with one fixed amount impact to track the sessions. A scaled dollar impact is not necessary because there is no charge for the first 10 sessions.

Figure 15–11 Quantity Discount Bracket 1

	Resource ID	GLID	Impact Categ...	S	P	D	Fixed Amount	Scaled Amou...	Units
1	IP sessions [1000125]	undefin...		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	1.00	0.00	None [0]

Figure 15–12 shows the second discount bracket.

The second bracket “10.00 - No Maximum” requires two balance impacts; one to charge the customer \$0.10 per minute and one to track IP sessions.

Figure 15–12 Quantity Discount Bracket 2

	Resource ID	GLID	Impact Categ...	S	P	D	Fixed Amount	Scaled Amou...	Units
1	US Dollar [840]			<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	0.00	0.10	Minute...
2	IP sessions [1000125]			<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	1.00	0	None [0]

Example 4: Rating Based on a Resource Balance Not Affected by the Rated Event

You can charge customers based on resources that do not impact a customer’s balance. For example, you can price IP usage based on the total number of employees in a company, or the net worth of a company. These resources are not directly related to the event you are rating. For example, if a company has 100 employees or less, you could charge them \$0.03 per IP session; if they have more than 100 employees, you could charge them \$0.01 per IP session.

Important: Because the quantity discount brackets are based on a constant resource value, no upper boundary is included in the range. This is different from all other types of rating. In this example, the number of employees is independent of the number of IP sessions.

Figure 15–13 shows how to set up the brackets in Pricing Center:

Figure 15–13 Rate Structure Based on Number of Employees

Notice that the maximum value for the first bracket is 101 and the minimum value for the second bracket is 101. The maximum value is not included in the valid range and the minimum value is included.

Example 5: Rating an Event As If Using a Single Balance Impact Group

When you use a rate with multiple quantity discount brackets, which bracket is in effect changes as the event quantity changes.

For example, suppose you have the brackets to rate 15 hours of IP usage as shown in [Table 15–3](#):

Table 15–3 Discount Brackets and Rating

Quantity Discount	Event Quantity	Fixed Amount	Scaled Amount	Result of Rating
Bracket 1: 0-5 hours	5	0	\$1	\$5.00
Bracket 2: 5-12 hours	7	0	\$0.75	\$5.25
Bracket 3: > 12 hours	3	0	\$0.55	\$1.65
NA	NA	NA	Total	\$11.90

Note: Because these fees are assessed on a per-hour basis, each bracket uses only a scaled amount.

Suppose, however, that you want to charge your customer as if all 15 hours were rated by the third group because *the total number of hours falls within that group's range*. In this case, the result is 12 hours @\$0.55/hour or a fee of \$8.25. To obtain this result, use fixed amounts in the second and third groups to compensate for what was charged in the

previous groups as shown in [Table 15-4](#):

Table 15-4 Rating Example

Quantity Discount	Event Quantity	Fixed Amount	Scaled Amount	Result of Rating
Bracket 1: 0-5 hours	5	0	\$1	\$5.00
Bracket 2: 5-12 hours	7	-\$1.25	\$0.75	\$4.00
Bracket 3: > 12 hours	3	-\$2.40	\$0.55	-\$0.75
NA	NA	NA	Total	\$8.25

To determine the correct value for the fixed amount in each group, use the following formula:

$$F_c = -Q_p (S_p - S_c)$$

Where:

- F_c is the Fixed amount for current bracket.
- Q_p is the maximum value of the Quantity range from the preceding bracket.
- S_p is the Scaled amount from the preceding bracket.
- S_c is the Scaled amount from the current bracket.

For example, determine the fixed amounts for the scenario above as follows:

Quantity Discount Bracket 1

A fixed amount is unnecessary.

Quantity Discount Bracket 2

Scaled amount for 5-12 hours = .75

Use the formula above and values from the previous bracket to determine the correct fixed amount:

$$F_c = -5 (1 - .75)$$

$$F_c = -1.25$$

Quantity Discount Bracket 3

Scaled amount for more than 12 hours = .55

Use the formula above and values from the previous bracket to determine the correct fixed amount:

$$F_c = -12 (.75 - .55)$$

$$F_c = -2.4$$

Using Event Attributes to Rate Events in Real Time

This chapter describes how to define rates in your Oracle Communications Billing and Revenue Management (BRM) price list by using event attributes such as telephone call origins and destinations.

Before you read this chapter you should be familiar with rating and price lists. See ["About Creating a Price List"](#).

About Using Event Attributes to Rate Events

As explained in ["About Creating a Price List"](#), if BRM can measure a billable event, it can rate it. It can rate an event based on multiple event attributes, such as the date and time that the event occurred. With attribute-based rating, you can specify additional event attributes to consider for rating, for example, phone call origin and destination.

Note: You can use a rate plan selector to rate cycle events. To do so, you can use account or service attributes, but not event attributes, to select the rate plan.

Using Event Attributes to Select Rate Plans

To use attribute-based rating, you use the Pricing Center *rate plan selector*. The rate plan selector associates a value in an event attribute with a rate plan. For example, by using the event attribute that stores the telephone call destination, you can choose rate plans like this:

- If the call is made to France, use the Calls to Europe rate plan.
- If the call is made to Ghana, use the Calls to Africa rate plan.

[Figure 16-1](#) shows a rate plan selector that chooses a different rate plan depending on call type. In this case, the telephony event includes an attribute named `CALL_TYPE`, which specifies the type of call, for example, a call made with a calling card. To rate a call made with a calling card differently from a call without a calling card, you assign a different rate plan based on the call type.

Figure 16-1 Rate Plan Selector Configuration for Call Type

Row	EVENT.PIN_FLD	CALL.PIN_FLD	TYPE	+	Rate Plan	Impact Category
1	5				Calling card	default
2	1				Home phone	default

The event attributes that you use for rating often depend on the source of the usage data. For example, when rating telephone calls, some PBX-to-gateway/gatekeeper systems do not supply the ANI value. In that case, you can use the CALL.ORIG value instead.

Using Event Attributes to Define Impact Categories

You use impact categories to apply different balance impacts for the same rate plan. For example, you can assign a set of telephone calls and destinations to a single rate plan:

Rate plan: IP Telephony

Impact category: USA to Europe

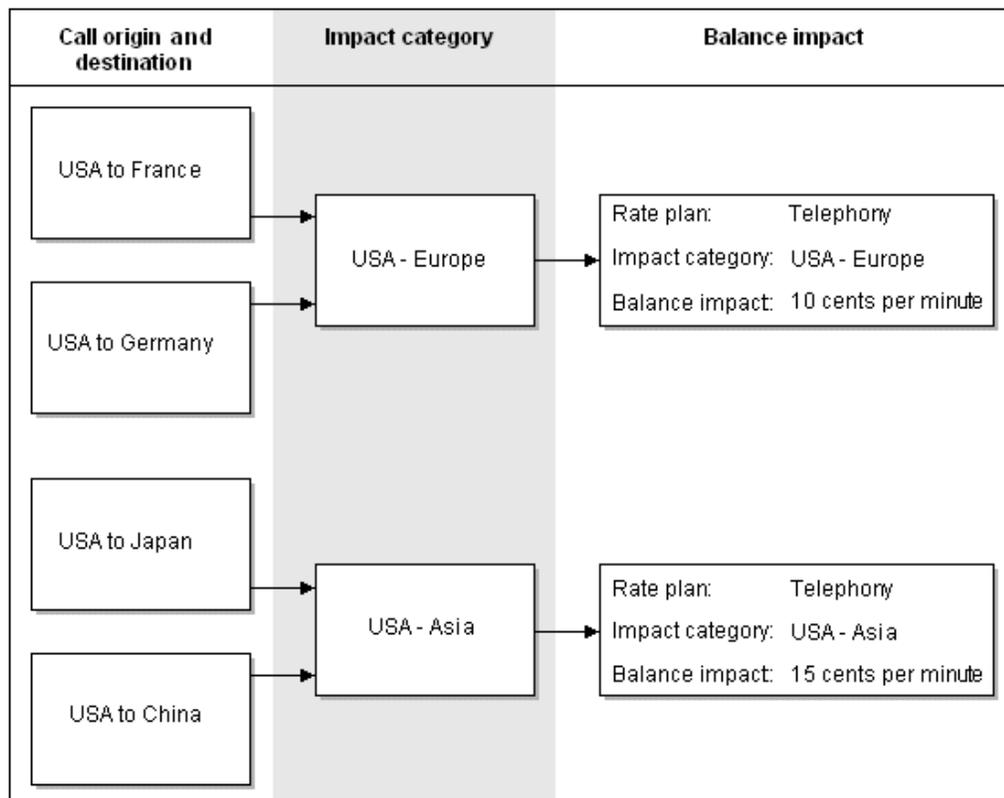
Balance impact: .10

Impact category: USA to Asia

Balance impact: .15

In [Figure 16–2](#), calls made to Europe are assigned to the USA - Europe impact category, and calls made to Asia are assigned to the USA - Asia impact category:

Figure 16–2 Call Impact Categories



Notice that the same rate plan is used for all calls, but the impact category specifies different balance impacts.

Even though you are using a single rate plan here, you must use the rate plan selector to assign impact categories. Therefore, you must choose **Rate plan selector** when creating the product, not **Single rate plan**.

You use the rate plan selector to associate event attributes with impact categories. The event attributes and impact categories shown in the figure above would look like [Figure 16-3](#) in the rate plan selector:

Figure 16-3 Rate Plan Selector

Row	EVENT.PIN_FLD_CALL.PIN_FLD_ANI	EVENT.PIN_FLD_CALL.PIN_FLD_DNIS	Rate Plan	Impact Category
1	1	33	IP Telephony ...	US_to_Europe
2	1	49	IP Telephony ...	US_to_Europe
3	1	81	IP Telephony ...	US_to_Asia
4	1	86	IP Telephony ...	US_to_Asia

The event attributes ANI (call origin) and DNIS (call destination) are entered as country codes:

- 1 = USA
- 33 = France
- 49 = Germany
- 81 = Japan
- 86 = China

A call from the USA to France might have this call origin and destination:

- Origin: 15555121212
- Destination: 3372621234

BRM matches the numbers in the rate plan selector with the telephone numbers. Since only country codes are used, all calls from each country match the numbers in the rate plan selector. For example, "3372621234" matches "33."

In the balance impacts for this rate plan, the impact categories define how much to charge as shown in [Figure 16-4](#):

Figure 16-4 Balance Impacts Based on Impact Category

Balance Impacts									
Legend									
P - Proratable D - Discountable S - Sponsorable									
Add Delete Print... Zoom...									
	Resource ID	GLID	Impact Category	S	P	D	Fixed Amount	Scaled Amount	Units
1	US Dollar [840]	Dialup Usage ...	US_to_Europe	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0.00	0.10	Minute [2]
2	US Dollar [840]	Dialup Usage ...	US_to_Asia	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0.00	0.15	Minute [2]

Note: In **Balance Impacts**, you can use asterisk (*) as a wildcard character in the **Impact Category** column to apply the balance impact to all impact categories defined for the rate plan.

Using the above examples, here is how BRM assigns a balance impact to a call:

1. The customer makes a call from the USA to France.
2. During rating, BRM finds the call origin and destination data in the event.

3. BRM looks through the data in the rate plan selector to find entries that have a call origin of USA and a call destination of France. This information tells BRM which rate plan to use, and which impact category to use.
4. BRM looks in the IP Telephony rate plan for the balance impact.
5. BRM looks through the IP Telephony balance impacts to find one that uses the USA_Europe impact category. BRM uses that balance impact to calculate the charge for the call.

About the Default Impact Category

You use the default impact category to apply a balance impact when none of the other impact categories can be used. The default impact category provides a fail-safe; without it, some events might not be rated at all.

The default impact category name is **default**.

Setting the impact category to **default** applies the balance impacts of the rate plan according to standard rating attributes, and ignores all impact category conditions.

Creating Impact Categories

To make impact categories available to Pricing Center, you edit the **pin_impact_category** file, then run the **load_pin_impact_category** utility to load the contents of the file into the **/config/impact_category** object in the BRM database.

Important: The **load_pin_impact_category** utility requires a configuration (**pin.conf**) files.

1. Edit the **pin_impact_category** file in *BRM_Home/sys/data/config*. The **pin_impact_category** file includes examples and instructions.

Caution: The **load_pin_impact_category** utility overwrites existing impact categories. If you are updating impact categories, you cannot load new impact categories only. You must load a complete set of impact categories each time you run the **load_pin_impact_category** utility.

2. Save the **pin_impact_category** file.
3. Use the following command to load the file into the database:

```
load_pin_impact_category pin_impact_category
```

If you are not in the same directory as the **pin_impact_category** file, include the complete path to the file, for example:

```
load_pin_impact_category BRM_Home/sys/data/config/pin_impact_category
```

For more information, see "[load_pin_impact_category](#)".

4. Stop and restart the Connection Manager (CM). If necessary, stop and restart Pricing Center.

To verify that the impact categories were loaded, you can display the **/config/impact_category** object by using the Object Browser, or use the **robj** command with the **testnap** utility.

Creating Unassigned Impact Categories

You can include more impact categories in the `pin_impact_category` file than you need. You do not have to assign a balance impact to all impact categories.

For example, you can plan ahead for origin/destination pairs that you will add later and create the impact categories that they will use.

About Charging for Custom Events and Attributes

BRM supports a default set of event attributes, such as geographical zones, quality of service, and so on. You can define charges for these attributes by using the Rate Plan Selector.

If you want to rate custom attributes of events or non-event attributes or if your rating guidelines are more complex than the defaults provided with BRM, you must modify the `PCM_OP_ACT_POL_SPEC_RATES` policy opcode.

For example, you can rate administrative events or non-event attributes such as an account attribute or a profile attribute. You can modify the policy to rate percentage values or values containing a greater than or less than operator. You can rate an event one way if an attribute is greater than 10, and another way if it is less than 10.

Charging for Custom Events and Attributes

To charge for custom events and attributes:

1. Modify the `PCM_OP_ACT_POL_SPEC_RATES` policy opcode or write a custom policy opcode to analyze the event data and assign the rate plan and impact category.

See the policy source file of the `PCM_OP_ACT_POL_SPEC_RATES` policy opcode for a sample implementation.

For information on how generic rating works, see the description of `PCM_OP_ACT_USAGE`.

2. Edit the `pin_spec_rates` file in `BRM_Home/sys/data/config` to associate the opcode to the event type that it rates.

The `pin_spec_rates` file includes examples and instructions.

Caution: When you run the `load_pin_spec_rates` utility, it overwrites the existing setup for administrative events charges. If you are updating a set of administrative events charges, you cannot load new charges only. You load complete sets of charges each time you run `load_pin_spec_rates`.

3. Create a configuration file for the "`load_pin_spec_rates`" utility.
4. Run the `load_pin_spec_rates` utility to load the contents of the `pin_spec_rates` file into the database.

```
load_pin_spec_rates pin_spec_rates_file
```

5. Edit the `pin_impact_category` file in the `BRM_Home/sys/data/config` directory to define impact categories for the event.

The `pin_impact_category` file includes examples and instructions.

Caution: When you run the `load_pin_impact_category`, it overwrites the existing impact categories. If you are updating a set of impact categories, you cannot load new impact categories only. You load complete sets of impact categories each time you run the `load_pin_impact_category` utility.

6. Run the "`load_pin_impact_category`" utility to load the impact categories into the database.

`load_pin_impact_category pin_impact_category_file`

7. If you use pipeline batch rating, add the custom cycle event to the following locations:
 - The EAI payload configuration file. See "About the Payload Configuration File" in *BRM Installation Guide*.
 - Your system's event notification list. The information you must add to the list is in the `pin_notify_ifw_sync` file.
8. In Pricing Center, create the rate plans that can be used to rate an event that uses custom event analysis.
9. In your pricing plan, assign a rate for the impact category.
10. Use the Pricing Center to create products that use the rates defined in your custom opcodes. Ensure that the products use custom event analysis.

Using Custom Fields with Real-Time Rating

If you add rating-related custom fields to BRM, you must make these fields available to the rating opcodes. For instructions, see "Making Custom Fields Available to Your Applications" in *BRM Developer's Guide*.

About Using Zones to Group Event Attributes

You use zones to group event attribute values into manageable categories. For example, to apply the same rate plan or impact category to all area codes in Northern California, you can create a zone that includes all Northern California area codes. Another zone might include all Southern California area codes. You then might include both zones in a zone for all California area codes, as shown in the following example:

California zone

Northern California

408

415

650

Southern California

213

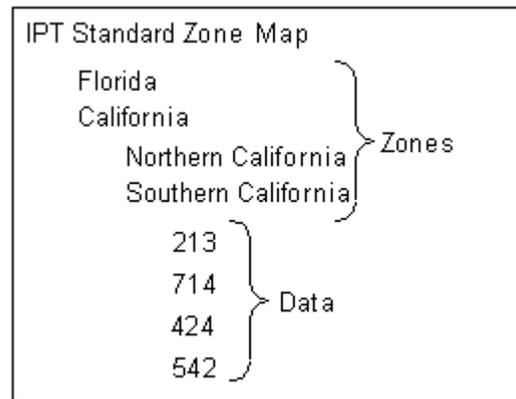
714

424

A zone map contains a hierarchy of zones and *data*, such as phone numbers, IP addresses, and Quality of Service (QoS) data.

Figure 16–5 shows a zone map with multiple nested zones. The data consists of area codes:

Figure 16–5 Zone Map with Nested Zones



If you use a zone map for an event attribute, all of that type of event attribute must be defined with a zone or data from that zone map. You cannot mix hard-coded values in a rate plan selector that uses zones.

For more information on zone and zone maps, see ["About Zoning"](#) and ["About Zone Models"](#).

About Setting Up Zones

When you use zone maps to rate events, BRM looks for the product to use for rating an event. If the product is configured to use a pipeline zone model, the real-time rating opcodes send the event to a real-time zoning pipeline to retrieve the zoning information. If the product is not configured to use a pipeline zone model, the real-time rating opcodes search the BRM database for the zoning information for the event.

You can define zones to rate events by using Pricing Center or Zone Mapper:

- To use zone information for rating both real-time and batch rating, use Pricing Center. Zone maps defined using Pricing Center are stored in the Pipeline Manager database.

If you use both real-time and batch processes for rating events, or if you are a new BRM user, define your zones by using Pricing Center.

See ["About Zoning"](#) and ["Setting Up Zones by Using Pricing Center"](#).

- To use zone information for rating only real-time events, use Zone Mapper in Pricing Center. Zone maps defined by using Zone Mapper are stored in the BRM database.

If you have already defined zone maps in BRM by using Zone Mapper, you can continue to use your existing zone maps.

See ["About Setting Up Zones by Using Zone Mapper"](#).

Pricing Center displays the zone-map names and impact categories from both the Pipeline Manager database and the BRM database. You can use the zone maps stored in either database to configure price plans.

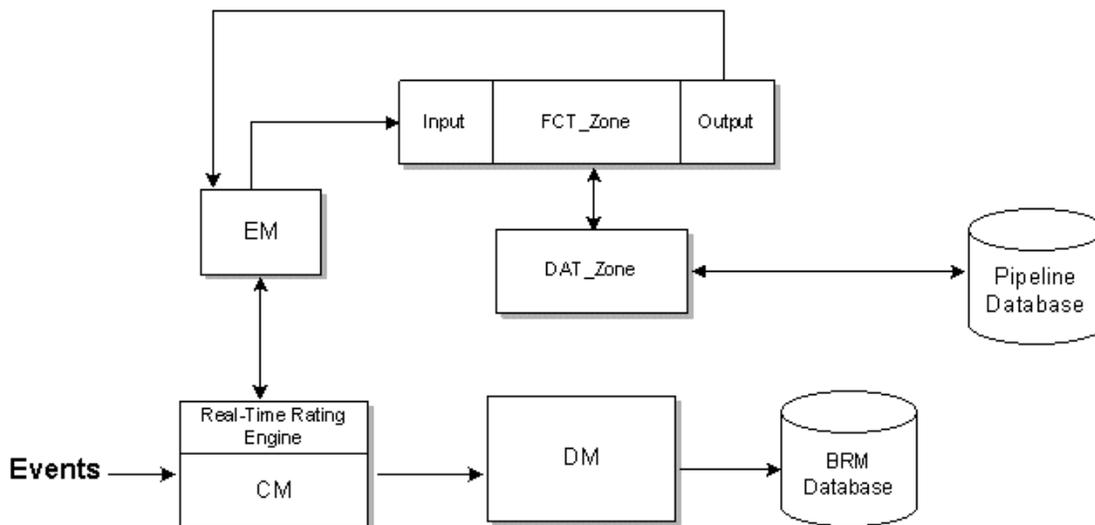
Real-Time Zoning Pipeline Architecture

A real-time zoning pipeline processes events as follows:

1. When the CM receives an event with a zoning request, the real-time rating opcodes check the event and the product that rates the event.
2. If the product is configured for pipeline zoning, the real-time rating opcodes extracts the calling number, called number, service calls, event time, and zone map name from the event and product flist.
3. BRM sends the event flist in a request to the NET_EM module to get the impact category from the Pipeline Manager database for the event. The CM sends the request for specific zone map data, such as standard zone-map data and geographic zone-map data. See "Configuring the NET_EM Module for Real-Time Processing" in *BRM System Administrator's Guide*.
4. When an event flist is passed to a real-time zoning pipeline, the input module in the pipeline converts the event data from the flist format used by real-time rating opcodes to the EDR format used in the pipeline.
5. The FCT_Zone module processes the zoning EDR and uses the DAT_Zone module to search for the zone map for the event.
6. When zoning is complete, the EDR, now containing zoning information for the event, is converted back into flist format by the output module in the pipeline. The output module also passes the event back to the real-time rating opcodes.
7. If a correct zone map for the event is not found in the Pipeline Manager database, the pipeline zoning module returns an empty flist to the CM. The CM logs the error and the event is not rated.

Figure 16–6 illustrates the data flow for real-time pipeline zoning:

Figure 16–6 Data Flow for Real-Time Pipeline Zoning



Setting Up Zones by Using Pricing Center

1. In Pricing Center, set up your zone models, which are stored in the Pipeline Manager database.

Important: Ensure that the zone map names are unique between both the BRM and Pipeline Manager databases.

See "[Setting Up Zone Models](#)".

2. Configure the NET_EM to receive events from the real-time rating opcodes.
3. Configure the CM to point to the NET_EM.
4. Configure the INP_Realtime module.

Use this entry for the OpcodeMapping entry:

```
OpcodeMapping = ./formatDesc/Formats/Realtime/zonemap_event.xml
```

5. Configure the OUT_Realtime module.
6. Configure the FCT_Zone module to specify a connection to the DAT_Zone module.
7. Configure the DAT_Zone module to process real-time events.

About Setting Up Zones by Using Zone Mapper

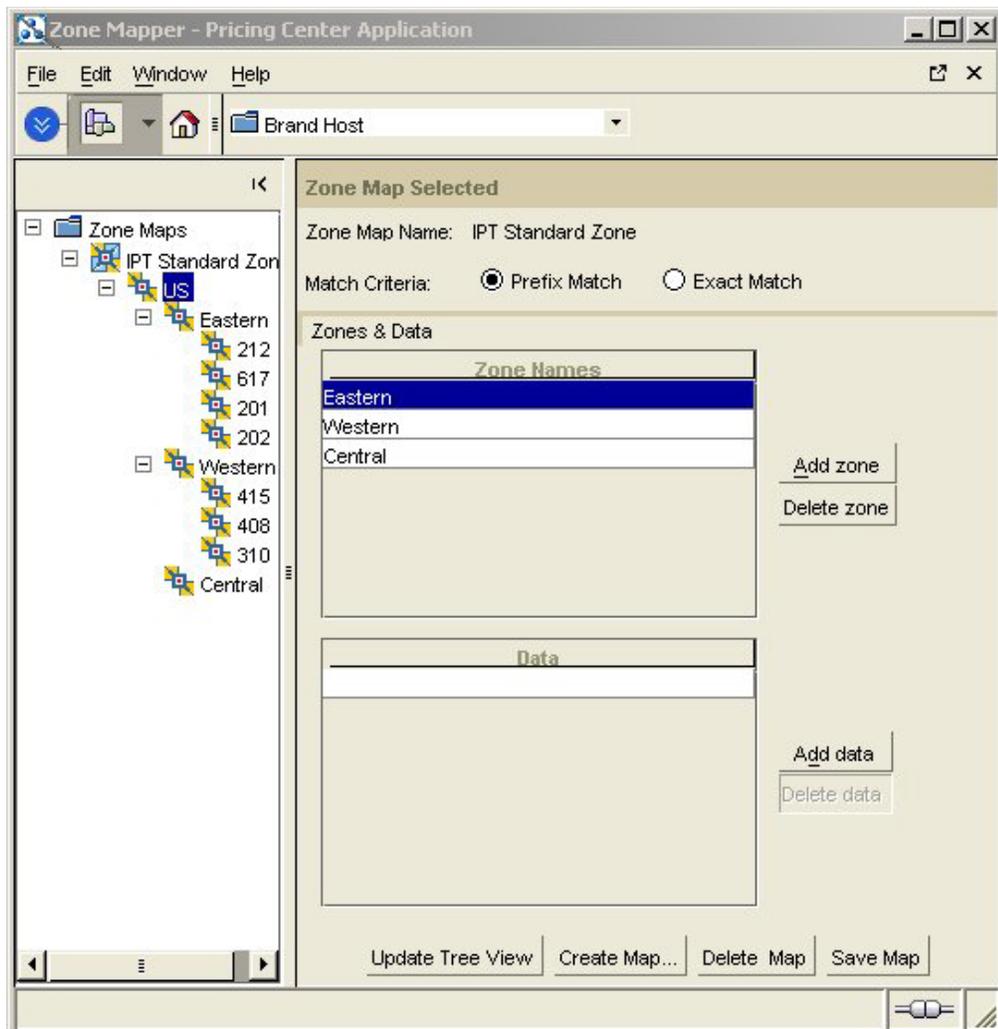
You use Zone Mapper to create zones and zone maps, which are stored in the BRM database. Zone Mapper is a component of Pricing Center. You can create multiple zone maps.

Important: You can use a rate plan selector to rate cycle events. To do so, you can use account or service attributes, but not event attributes, to select the rate plan.

Note: A zone map name can include a maximum of 255 characters.

[Figure 16-7](#) shows Zone Mapper.

Figure 16–7 Pricing Center Zone Mapper



About Defining Zone Maps for Products

Each product can have only one zone map. However, one zone map can be used by multiple products. You can create multiple zone maps for one system or for branded systems.

Note: All brands in a database can use any of the zone maps stored in that database.

Options for Matching Event Attribute Values

When you create zones using Zone Mapper, you specify how BRM matches a value in an event with the data in the zone map. You have two choices:

- Prefix:** BRM searches for the value in the zone map that best matches the event. For example, in an IP telephony service, if the customer calls 202-555-5678, and the zone map includes 202, BRM will find the zone with 202. If the zone map includes both 202 and 202-555, BRM will find the zone that includes 202-555.

- **Exact Match:** BRM searches only for the value in the zone map that exactly matches the event. For example, if the customer calls 202-555-5678, BRM looks only for a zone map that includes 202-555-5678.

Changing and Deleting Zone Maps

You can either change a zone name, modify the data for a zone, or delete a zone map entirely. When you delete a zone map that you used to define a product in Pricing Center, you must also remove the map from that product. When you delete a zone map, you also delete the zones that belong to it.

Caution:

- Deleting a zone map permanently removes the map from the BRM database.
 - If customers have already purchased a product that includes a zone map that you are deleting, you must cancel the product that includes the old zone map, and purchase the product that includes the new zone map.
-
-

Selecting Multiple Rate Plans and Impact Categories

You can use the rate plan selector to choose rate plans and assign impact categories at the same time.

Figure 16–8 shows a rate plan selector that selects rate plans based on three event attributes:

- Call origin
- Call destination
- Call type

Figure 16–8 Rate Plan Selector

Row	Q_CA...	Q_CAL...	EVENT.PIN_FLD_CALL.PIN_FLD_TYPE	+	Rate Plan	Impact Category
1	US	Europe;France	5		Calling card	US_to_Europe
2	US	Europe;Germany	5		Calling card	US_to_Europe
3	US	Europe;France	1		Telephony	US_to_Europe
4	US	Europe;Germany	*		Telephony	US_to_Asia

Customizing Zone Mapping

The zone mapping opcodes perform zoning for real-time rating.

For background information, see ["About Using Zones to Group Event Attributes"](#).

How Zone Mapping Works

To manage zones, for example, create, delete, and update zone maps, use PCM_OP_ZONEMAP_COMMIT_ZONEMAP. Zone maps are stored in **/zonemap** objects.

By default, PCM_OP_ZONEMAP_COMMIT_ZONEMAP takes an array of zone maps as its input, each of which represents the new, modified, or deleted zone map. In the case of a deletion, the opcode marks the zone map to be deleted in the input flist.

The PCM_OP_ZONEMAP_COMMIT_ZONEMAP output is an flist that links zone maps and products.

To retrieve zone map information from the database, use PCM_OP_ZONEMAP_GET_ZONEMAP. The default implementation enables you to retrieve zone maps in several ways:

- You can use it get all the zone maps by providing a zone map type POID
- You can use it get a specific zone map by providing the zone map name
- You can get the zone map associated with a product by providing the valid product POID

PCM_OP_ZONEMAP_GET_ZONEMAP is called by the Zone Mapper to retrieve zone map data and displays it in the user interface. It is also called by the zone map search opcode, PCM_OP_ZONEMAP_POL_GET_LINEAGE.

Customizing Zone Mapping

Use the following opcodes to customize zone mapping:

- PCM_OP_ZONEMAP_POL_GET_LINEAGE. See ["Finding Zone Maps"](#).
- PCM_OP_ZONEMAP_POL_GET_ZONEMAP. See ["Getting Zone Maps from the BRM Database"](#).
- PCM_OP_ZONEMAP_POL_SET_ZONEMAP. See ["Saving Zone Map Data"](#).
- PCM_OP_RATE_POL_EVENT_ZONEMAP. See ["Getting Zone Maps and Impact Categories from the Pipeline Manager Database"](#).

Finding Zone Maps

To search in a zone map for data associated with a given string, use PCM_OP_ZONEMAP_POL_GET_LINEAGE.

You supply a string and a zone map name. It then searches the zone map for the given string and returns the matching node with all ancestors of the matching node (the *lineage*).

This opcode calls the zone-based rating load opcode, PCM_OP_ZONEMAP_GET_ZONEMAP.

Getting Zone Maps from the BRM Database

To customize which zone map data to retrieve from the database, use PCM_OP_ZONEMAP_POL_GET_ZONEMAP. This opcode provides zone maps to display in the Zone Mapper.

This opcode retrieves the original zone map data from the BRM database, and depending on the calling program, it converts the binary format of zone map data into a readable flist, or it returns the binary format to be used in the lineage search.

The input flist must include either the PIN_FLD_POID for routing purposes or PIN_FLD_NAME to search by name.

Saving Zone Map Data

To customize how to save zone map data, use PCM_OP_ZONEMAP_POL_SET_ZONEMAP. This opcode saves zone map in the BRM database when you commit zone maps in the Zone Mapper.

This opcode retrieves the original zone map data from the BRM database, and depending on the calling program, it converts the binary format of zone map data into a readable flist, or it returns the binary format to be used in the lineage search.

The input flist must include either the PIN_FLD_POID for gateway routing purposes or PIN_FLD_NAME to search by name.

Getting Zone Maps and Impact Categories from the Pipeline Manager Database

To get zoning data from the Pipeline Manager database, use the PCM_OP_RATE_POL_EVENT_ZONEMAP policy opcode. This opcode returns the zone map name and impact category for an event from the Pipeline Manager database.

You can customize this policy to add new event classes that you have created, if those event classes use a real-time zoning pipeline.

PCM_OP_RATE_POL_EVENT_ZONEMAP performs the following functions:

1. Reads the event storable class.
2. Based on the service, it retrieves the calling number, the called number, and the zone map name from the event.
3. Prepares the input flist for a real-time zoning pipeline input module.
4. If it finds a matching impact category for the event zone, returns the impact category for the event.
5. If it doesn't find a matching impact category, returns an empty string and logs an error.

PCM_OP_RATE_POL_EVENT_ZONEMAP returns the zone map name and impact category for the event. If the opcode fails, it returns an empty string and logs an error.

Using the Rate Plan Selector in Pricing Center

Before you use the rate plan selector, you must create impact categories. See "[Creating Impact Categories](#)".

When you use a rate plan selector in Pricing Center, you start with an empty rate plan selector as shown in [Figure 16-9](#):

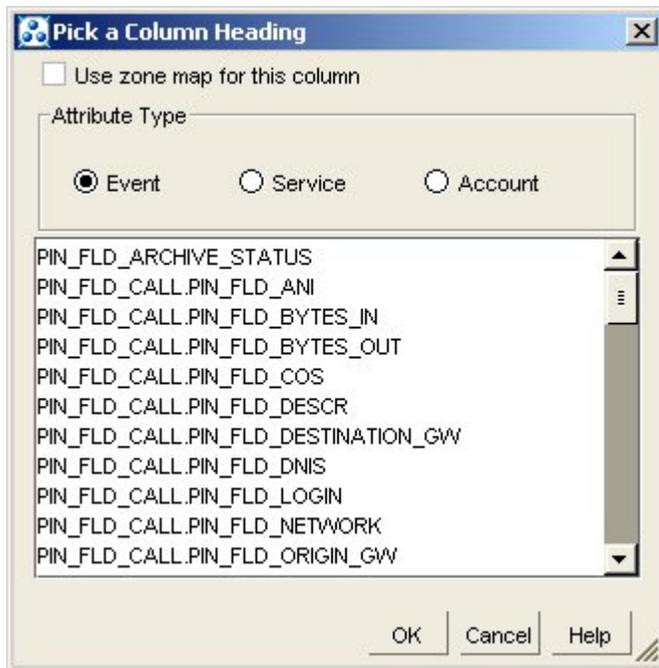
Figure 16-9 Empty Rate Plan Selector

The screenshot shows the 'Rate Plan Selector' window. At the top, there is a text input field labeled 'Plan selector name' containing the text 'New'. Below it is a dropdown menu labeled 'Zone map' with 'IPT Standard Zone' selected. A table is displayed below the dropdown, with columns labeled 'Row', '+', 'Rate Plan', and 'Impact Category'. The 'Row' column has a '+' sign in the first row. To the right of the table, there are several buttons: 'Edit Plans...', 'Move Up ^', 'Move Down v', 'Add Row', 'Delete Row', and 'Delete Column'.

After you provide a name for the rate plan selector, and (optionally) choose a zone map, you specify the attribute type and attributes you want to use.

To choose the attribute type, you click the appropriate option. To choose attributes, you select from a list of all attributes in the object. [Figure 16–10](#) shows the list for an IP dialup session event:

Figure 16–10 IP Dial Up Session Attributes in Rate Plan Selector Configuration



To add rate plans and impact categories, you choose from lists. You can create a rate plan if it is not on the list, but impact categories must already be defined.

Set the impact category to **default** to apply the balance impacts of the rate plan according to standard rating attributes, and ignore all impact category conditions.

Using Wildcards to Match Event Data

In a rate plan selector, use an asterisk (*) as a wildcard as follows:

- In attribute columns, enter an asterisk to match any value in an event.
- In the **Impact Category** column, do not use an asterisk. BRM does not support its use as a wildcard in this column.

See "[Using Event Attributes to Define Impact Categories](#)" for information about using wildcard characters in rate plan balance impacts.

Specifying the Priority for Attributes

The order of the rows in the rate plan selector determines the order in which rate plans are processed. The top row has the top priority. For example, you can specify to use one rate plan for one value of an attribute, and another rate plan for all other values. To do this, you give the value the top priority as shown in [Figure 16–11](#):

Figure 16–11 Rate Plan Selector

Row	EVENT.PIN_FLD_CALL.PIN_FLD_TYPE	+	Rate Plan	Impact Category
1	5		Calling card	default
2	*		Non-card	default

When you use multiple attributes, they are processed in the order from left to right.

About Pipeline Rate Plans

This chapter describes how to configure rate plans for rating with Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager. It includes information about using Pricing Center to create rate plans and price models.

Before you read this chapter, you should be familiar with these topics:

- Pipeline rating. See "About pipeline rating" in *BRM Configuring Pipeline Rating and Discounting*.
- Real-time rating. See "[About Real-Time Rate Plans](#)".

About Configuring Pipeline Rating

Configuring pipeline rating involves two sets of tasks:

- Creating rate plans, price models, and other data using Pricing Center. See "[Creating Pipeline Rate Plans and Price Models](#)".
- Configuring rating function modules. See "Configuring function modules for pipeline rating" in *BRM Configuring Pipeline Rating and Discounting*.

Creating Pipeline Rate Plans and Price Models

Pipeline rate plans define the criteria used to determine how EDRs processed by a pipeline are charged. Rate plans also include price models that define the actual charges. For example, a rate plan can define that the charge for a friends and family call from the US to the UK at 8 p.m. local time is 9 cents per minute or that SMS messages cost 15 cents no matter what time they are sent.

Price model selectors contain multiple price models. Each price model is associated with rules based on EDR values. When a price model selector is used in a rate plan, Pipeline Manager uses the EDR values to choose a price model for an event.

Pipeline rate plans and price models are used by the FCT_MainRating module to apply charges to an EDR in a pipeline.

This section provides an overview of pipeline rate plans and price models. For step-by-step instructions about using Pricing Center, see Pricing Center Help.

Understanding the Difference between Real-Time and Pipeline Rate Plans

There are two separate but related kinds of rate plans in BRM. In the simplest terms, *real-time rate plans* are used for real-time rating while *pipeline rate plans* are used for pipeline rating. However, both real-time and pipeline rate plans are necessary for pipeline rating.

During pipeline rating, the name of a real-time rate plan is added to the EDR by the FCT_CustomerRating module. This module determines the correct real-time rate plan based on information in the account associated with the EDR or from information in the EDR itself.

As rating continues, the FCT_MainRating module uses the name of the real-time rate plan in the EDR to select a pipeline rate plan. Each pipeline rate plan has a unique code that corresponds to the name of a real-time rate plan. So if the FCT_CustomerRating module adds the real-time rate plan **Corporate** to an EDR based on a service-level rate plan ERA, this rate plan name is used to select the pipeline rate plan that includes the code **Corporate**. A version and configuration of this pipeline rate plan is then used to perform the actual rating.

Note: In general, you do not need to concern yourself with real-time rate plans when you work on pipeline rating. Real-time rate plans matching pipeline rate plans are automatically created when you use Pricing Center to associate a pipeline rate plan with an event.

About Pipeline Rate Plans

A rate plan has one or more rate plan versions. Each version contains rate plan configurations that determine the price model to use for the event being rated.

The rate plan itself contains data that applies to all versions and configurations, including the rate plan name and code, status, model type, splitting type, calendar, time offset, currency, and tax treatment. For details about using Pricing Center to create rate plans, see Pricing Center Help.

Rate plans are stored in the IFW_RATEPLAN table in the Pipeline Manager database.

See ["About Pipeline Rate Plan Versions"](#) and ["About Pipeline Rate Plan Configurations"](#) for more information about these components of the rate plan. See ["About Pipeline Rate Adjustments"](#) for information about optional rate adjustments.

Data Required to Create Rate Plans

You define the following kinds of data to define rate plans:

- Currencies. See ["Defining Currencies"](#).
- Impact categories. See ["About Impact Categories"](#).
- Price models. See ["About Pipeline Price Models"](#).
- (Optional) Price model selectors. See ["About Price Model Selectors"](#).
- Service codes and classes. See ["About Mapping Services"](#).
- Time models. See ["Rating by Date and Time with Pipeline Manager"](#).
- Zone models. See ["About Zoning"](#).

Rating EDRs Split across Time Periods

EDRs 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. When you define a rate plan, you specify a Splitting option that determines how an EDR that overlaps a time period is rated.

You can choose from four splitting options:

- No splitting, based on start time: Use the start time to rate the entire call.

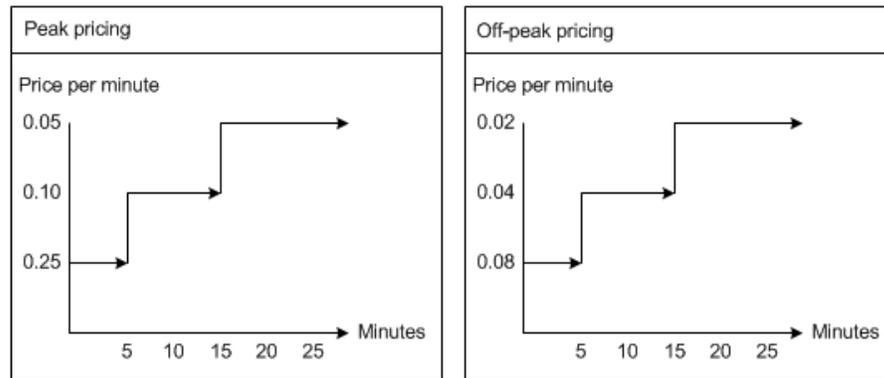
- No splitting, based on end time: Use the end time to rate the entire call.
- Consecutive splitting: Split the call into separately rated parts. When using multiple price steps for different usage levels, continue counting the call duration from the time of the split.
- Isolated splitting: Split the call into separately rated parts. When using multiple price steps for different usage levels, start over at zero from the time of the split.

The Consecutive and Isolated splitting options are used when you use a combination of price steps and time-based rating. With this combination, you can have a case where an event is split by time zones, resulting in the application of two different Price Models, each with its own Price Model Steps. The Consecutive and Isolated splitting options enable you to determine how to handle the selection of the appropriate Price Model Step.

Here is an example:

Figure 17-1 shows the peak pricing and the off-peak pricing:

Figure 17-1 Peak and Off-Peak Pricing



06:00 - 07:30 is peak

07:30 - 10:00 is off-peak

An EDR is rated that starts at 7:10 and finishes 7:35, for a total of 25 minutes. The event is split into two segments:

07:10 – 07:30 - peak (20 minutes)

07:30 – 07:35 - off-peak (5 minutes)

Figure 17-2 shows how Consecutive splitting works. The gray areas show how the Price Model Steps are applied in both peak and off-peak Price Models. In this case, the first 20 minutes apply to the peak Price Model, so the call is rated at .25 per minute, .10 per minute, and .05 per minute. The last 5 minutes are rated by the off-peak Price Model, which, since it uses Consecutive splitting, takes into account the previous 20 minutes, and rates the final 5 minutes at .02 per minute:

Figure 17-2 Consecutive Splitting

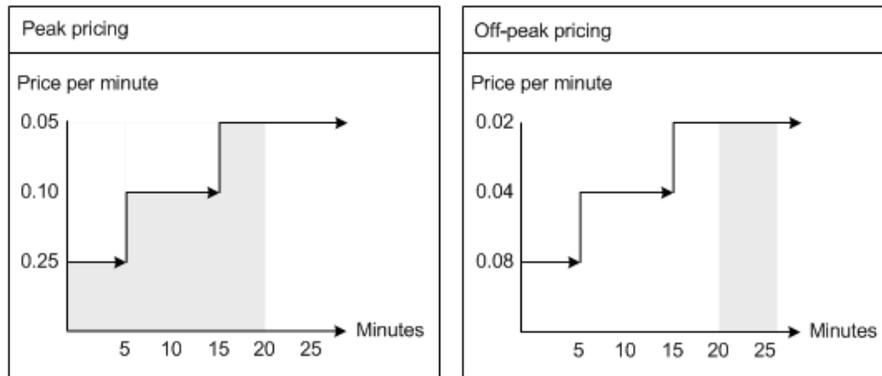
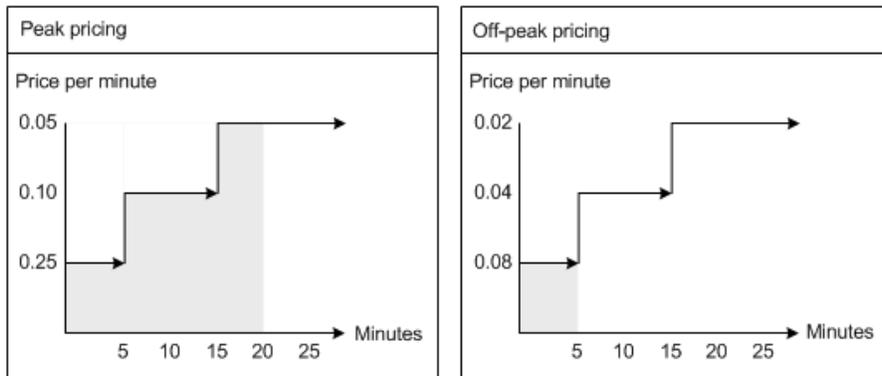


Figure 17-3 shows how Isolated splitting works. In this case, the first 20 minutes apply to the peak Price Model in the same way as Consecutive splitting. The last 5 minutes are rated by the off-peak model, which, because it uses Isolated splitting, does not consider the previous 20 minutes. Instead, it begins counting at zero, so the remaining 5 minutes are rated by the first step, that is, .08 per minute:

Figure 17-3 Isolated Splitting



Using Consecutive splitting and Isolated splitting gives different results for the final charge for the EDR. In this example:

Table 17-1 shows the results of **Consecutive splitting**:

Table 17-1 Results of Consecutive Splitting

Usage	Calculation	Result
Peak (20 minutes)	$(5 * 0.25) + (15 * 0.10) + (5 * 0.05)$	3.00
Off-peak (5 minutes)	$5 * 0.02$	0.10
NA	Total	3.10

Table 17-2 shows the results of **Isolated splitting**:

Table 17–2 Results of Isolated Splitting

Usage	Calculation	Result
Peak (20 minutes)	$(5 * 0.25) + (15 * 0.10) + (5 * 0.05)$	3.00
Off-peak (5 minutes)	$5 * 0.08$	0.40
NA	Total	3.40

About Pipeline Rate Plan Versions

Each rate plan must have at least one rate plan version and can have as many versions as you need. Each rate plan version is valid for a different time period and only one version is valid for each period. The start time of an EDR determines which rate plan version is used for that record.

In addition to the validity period, you specify a zone model when you define a rate plan version. The zone model determines how calls to and from different regions are classified for rating. Each rate plan version can use different zone models.

Rate plan versions are stored in the IFW_RATEPLAN_VER table in the Pipeline Manager database.

About Basic and Delta Rate Plan Versions

Rate plan versions can be either basic or delta versions.

- A *basic version* can be used as the basis of other versions. You must specify all required attributes when you create a basic rate plan version.
- A *delta version* inherits attributes from a basic rate plan version that you specify when you create the delta version. In the delta version, you enter only the attributes that you want to change. For example, you might change only the **Version**, **Valid From**, and **Zone Model** fields.

About Pipeline Rate Plan Configurations

The rate plan configuration determines which price model or price model selector is used to charge a given EDR. When you define a rate plan configuration, you specify a combination of criteria that an EDR must match to be rated by a particular price model.

Each configuration maps a combination of service code, service class, and impact category to a combination of time model and time period. The configuration then maps the time model/time period combination to a price model or price model selector.

You can create any number of configurations for a rate plan version. The configurations in a given rate plan version must cumulatively cover all possible combinations of service code, service class, impact category, time model, and time period.

When you associate a price model with a rate plan configuration, you can optionally specify an alternative price model that is used in addition to the main price model. You can compare the charges that result from the two models. For example, you may want to better understand the financial impact of a change to a different price model before committing to the change.

In addition, you can choose to replace or modify the calculated charge by a value that you enter. See "[Using Passthrough Prices](#)".

Rate plan configurations are stored in the IFW_RATEPLAN_CNF table in the Pipeline Manager database.

Using Passthrough Prices

When you define a rate plan configuration, you can choose to ignore the calculated price and use a price that is passed in by the CDR instead. For example, you can ignore the calculated charge and use a passed-in charge if you receive external wholesale charges and want to use them for retail rating.

You can also modify or replace the passed-in price by specifying an add-on type and entering a charge. There are three add-on types:

- **Percentage** increases the passed-in price by a percentage that you enter.
- **Addon Value** increases the passed-in price by a fixed amount that you enter.
- **New Value** replaces the passed-in price with an amount you enter.

To use the passed-in price without modification, specify **0** as the charge.

About Pipeline Rate Adjustments

Rate adjustments are an optional way to customize a pipeline rate plan version. You can use rate adjustments to provide discounts based on date, time, service, and other event attributes. For example, you can provide a discount on all calls for a specific day.

An adjustment can be a percentage change to the original charge, a value to be added to the charge, or a completely new value to replace the charge.

When you define a rate adjustment, you specify dates and times during which the adjustment is valid and a maximum quantity above which the adjustment does not apply.

You also specify rules that determine whether an EDR qualifies for the adjustment. These rules filter EDRs based on values such as usage class, usage type, service code, service class, impact category, source network, and destination network. You can enter fixed values, expressions, or a wildcard (.*) that matches all values.

When you use a rate adjustment, the original charge is overwritten by the adjusted charge. This is different from discounting, which leaves the original charge in place while calculating a discounted charge. Adjustments and discounts are also handled differently for accounting purposes. When calculating the general ledger (G/L) impact, the adjusted amount is not considered revenue. When you use discounting, however, the discounted amount is counted as revenue.

There are two ways to define rate adjustments:

- You can define rate adjustments in Pricing Center. In this case, the rate adjustment data is stored in the IFW_RATEADJUST table in the Pipeline Manager database.
- You can create a file that defines rate adjustment rules. For information on creating the file, see "Creating a rate adjustment rules file" in *BRM Configuring Pipeline Rating and Discounting*.

Rate adjustment is performed by the FCT_RateAdjust module. The module reads data from the EDR and evaluates it according to the rate adjustment rules stored in the database or in the rate adjustment file.

About Pipeline Price Models

Price models define the charges that apply to an EDR when it is rated by a pipeline. The rate plan configuration determines which price model is used for an EDR.

Price models can be divided into steps that define different prices for different usage levels. For example, you can charge a customer ten cents per minute for the first five minutes and eight cents per minute for usage after that. A price model must have at least one step, but can have any number of additional steps. If there is only one step, it determines pricing for all usage levels.

Price model steps define how usage is measured (the RUM, beat, and charge basis) and the resource that is impacted by the charge. They can also include a minimum charge. See "[Setting a Minimum Charge](#)".

In the case of multiple resources, pipeline rating is done differently. For example, consider a price model with two price model steps, where both the resources rate on **Duration RUM**:

- Currency resource: \$0.1/min
- Resource voucher points: 0.5 point/min

Now, for an event with a one-hour duration (3600 seconds), the rating results are as follows:

- \$6
- 30 voucher points

In other words, the event is rated twice, once for the currency resource and then for the resource voucher points.

One usage event can impact more than one resource and consume more than one RUM. For example, a phone call can be measured in both minutes and volume of data. Similarly, it can impact both a monetary resource and a balance of bonus points. You must define pricing steps for all resource/ RUM combinations that have a balance impact. The same event is then charged using all the steps that apply to it.

For detailed instructions about using Pricing Center to create price models and price model steps, see Pricing Center Help.

Setting a Minimum Charge

When you create a price model step, you can specify a minimum charge. You can define separate minimums for each resource defined in the steps for a price model.

If you define different minimum charges in different price model steps for the same resource, the largest minimum charge is used during rating. For example, if one step defines a minimum charge of \$1.29 and another a minimum charge of \$0.99, \$1.29 is used.

When a minimum charge applies, the charge values for the affected resource in the EDR are overwritten and lost.

When an EDR includes multiple charge packets because of a time period split or multiple RUMs, the rating module checks to see if the total of the charge packets for a given resource in the EDR exceeds the minimum charge for that resource. If the total is *greater* than the minimum, the total is used. If the total of the charges is *less* than the minimum charge, the minimum charge is allocated among the charge packets so that their total equals the minimum charge.

Assigning G/L IDs in Price Models

You assign G/L IDs to price model steps. A price model that includes multiple steps can include different G/L IDs for each step if the steps use different RUMs.

About Price Model Selectors

A price model selector chooses a price model during rating based on values in an EDR. This enables you to apply different pricing to different scenarios in a single rate plan configuration. [Figure 17-4](#) shows the Price Model Selector Configuration window.

You can use price model selectors to apply preferential, promotional, or other selective pricing based on EDR characteristics.

The elements of a price model selector, and the relationship between them, are as follows:

- A *price model selector* contains one or more configurations. You rank the configurations in the order in which you want Pipeline Manager to evaluate them.
- A *configuration* maps a price/discount model selector rule to a price model. Each configuration has a validity period.

Figure 17-4 Price Model Selector Configuration



- A *price/discount model selector rule* associates an EDR field with a value. A rule can contain one or more such associations, all of which are logical and must be in the EDR for the rule to be satisfied. [Figure 17-5](#) shows a Model Select Rule configuration screen.

Figure 17-5 Price Model Selector Rule

Criteria For Applying		
	EDR Field Name	Criteria
<input type="checkbox"/>	DETAIL.CUST_A.ACTG_NEXT_DATE	30/04/2006 03:36:19
<input type="checkbox"/>	DETAIL.CUST_A.PRODUCT.PLAN_NAME	Standard

Note:

- You can use any EDR field that is defined in the data dictionary, except EDR fields that have the data format of Block.
- You use the same price/discount model selector rules for both discount model selectors and price model selectors.

Pipeline Manager evaluates the configurations in a price model selector in the order in which you rank them. When a rule in a configuration is satisfied (that is, when the specified EDR fields contain the specified values), Pipeline Manager uses the price model that is mapped to that rule and ignores any configurations that are ranked lower. If an EDR does not contain the specified values, it is not rated with the associated price model.

Important: When you set up price model selectors, the rules you use should ensure that all EDRs are rated. If an EDR does not meet one of the rules, it is not rated.

You set up price model selectors in Pricing Center.

Example of Using a Price Model Selector

In this example, a price model selector applies different price models based on the plan type, the service provider call type derived from the tariff class, and the sending carrier ID.

Following are the general steps for setting up this example:

1. Create two price models:
 - **PM.05_60:** The charge is 0.05 Euros, the beat is 60.
 - **PM.10_60:** The charge is 0.10 Euros, the beat is 60.
2. Create a price/discount model selector rule named RULE001 with the criteria as shown in [Table 17-3](#):

Table 17-3 Price Model Selector Rule001

EDR Field	Value
DETAIL.ASS_CBD.CP.RATEPLAN_TYPE	Standard
DETAIL.CALL_TYPE	CX_Call
DETAIL.CARRIER_ID	Carrier X

3. Create a price/discount model selector rule named RULE002 with the criteria as shown in [Table 17-4](#):

Table 17-4 Price Model Selector Rule002

EDR Field	Value
DETAIL.ASS_CBD.CP.RATEPLAN_TYPE	.*
DETAIL.CALL_TYPE	.*
DETAIL.CARRIER_ID	Carrier X

4. Create a price model selector with these combinations of price models and price/discount model selector rules as shown in [Table 17-5](#):

Table 17-5 Price Model and Rule Examples

Price Model	Rule
PM.05_60	RULE001
PM.10_60	RULE002

5. Create a rate plan with a configuration that uses the price model selector you created.
6. Create a product that will be rated with the rate plan you created.
7. Create a customer who purchases the product you created.

When an EDR for this customer/product combination is rated, the .05 Euro charge is used for a call from Carrier X with the plan Standard and the call type CX_Call. The .10 Euro charge is used for calls from Carrier X with other plans and call types.

Setting Up Pipeline Price List Data

This chapter describes how to configure price list data for the Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager.

Before reading this chapter, you should be familiar with BRM price lists and how Pipeline Manager works. See the following documents:

- [About Creating a Price List](#)
- "About Pipeline Rating" in *BRM Configuring Pipeline Rating and Discounting*

About Mapping Services

Incoming event data records (EDRs) from multiple switches often use different codes to represent the same service or supplementary service. To process EDRs, you must normalize that data by mapping external service codes to internal service codes, service classes, and usage classes. You also must map usage types so they can be added to EDRs.

- An internal *service code* represents a service such as voice or fax.
- An internal *service class* represents a variation on the service (for example, a GPRS quality of service (QoS) or a telephone service used only for emergency calls).
- An internal *usage class* represents a supplementary service (for example, call forwarding or a voice mail box). The data used for rating usage classes comes from the network.
- A *usage type* represents an attribute of a customer's account, such as an extended rating attributes (ERA), that is used in rating but which is not available from the network.

You define the internal codes and usage types in Pricing Center. All of these are used to rate usage.

To create internal service mappings, you use Pricing Center to set up the data that specifies how to map the external data to the internal codes. For example, you can specify that if the external service code is **011**, the internal service code is **TEL**.

The mapping data you enter in Pricing Center must match the syntax and format of the EDR input data.

About Creating Map Groups

You organize service codes, service classes, and usage classes in *map groups*. A map group specifies the set of services and usage scenarios to be used for rating by Pipeline

Manager. You specify the name of the map group when setting up the pipeline modules.

You can create any number of map groups. When you have multiple map groups, you can use a different set of service mappings with different pipelines. For example:

- You rate services that use various EDR formats by setting up a pipeline for each format.
- If you rate roaming usage, you might set up three pipelines using these service mapping scenarios:
 - A retail rating pipeline that uses service mappings and usage scenarios relevant to your customer’s accounts and services, such as friends-and-family calls, birthday calls, messaging services, and call forwarding.
 - Two pipelines (outcollect and incollect) for rating roaming usage that includes the services and usage scenarios described in your roaming agreements.

When you configure modules that use the map group, you enter the map group name in the registry. For example:

```
ServiceCodeMapping
{
  ModuleName = FCT_ServiceCodeMap
  Module
  {
    Active = True
    DataConnection = integrate.DataPool.Database
    MapGroup = serviceMapGroup
  }
}
```

Important: The map group name that you enter in the registry must exist in a map group configuration in the Pipeline Manager database.

Mapping Service Codes and Service Classes

To create service code and service class mappings, you use Pricing Center to specify which external data is used to determine the internal service codes and service classes. For example, you can specify that if the external service code is **011**, the internal service code is **TEL**.

The mappings are based on the following data:

- External service code
- Usage class
- Location area indicator
- VAS event product code
- Quality of service requested
- Quality of service used
- Record type

The mappings are stored in the IFW_SERVICE_MAP table. The mapping is performed by the FCT_ServiceCodeMap module.

In addition to mapping external service codes to internal service codes, you also map internal Pipeline Manager service codes to service types defined in the BRM database. These mappings are stored in the IFW_SERVICE database table.

The configuration in [Table 18–1](#) shows the mapping for three services:

Table 18–1 Service Mapping Configuration

Pipeline Manager Service Code	BRM Service Type
TEL	/service/telco/gsm/telephony
SMS	/service/telco/gsm/sms
GPRS	/service/ip/gprs

Multiple Pipeline Manager service codes can be mapped to a single BRM service type. For the best performance, use as few service mappings as possible. For example, map the TEL, SMS, DATA, and FAX Pipeline Manager service codes to a single /service/telco/gsm BRM service type.

About Setting Up Service Mapping

To set up service mapping, do the following:

1. Create internal service codes and service classes in Pricing Center.
2. Map the external data to the internal service codes and service classes in Pricing Center.
3. Configure the FCT_ServiceCodeMap module.

This module maps call data to service codes by using the data you entered in Pricing Center.

Mapping Events and Services

When you create your price list, the usage events that you specify must be associated with services in the Pipeline Manager IFW_REF_MAP table. This table specifies the service type to event type mapping so that the services in your price list are associated with usage events. In addition, the IFW_REF_MAP database table also specifies the prefix for the data that identifies an account (for example, MSISDN or IMSI). See ["Configuring Account ID Prefixes"](#).

You can map multiple Pipeline Manager service codes to a single BRM service type. For example, you could map the TEL, SMS, DATA, and FAX Pipeline Manager service codes to a single /service/telco/gsm BRM service type.

The data is stored in the Pipeline Manager database.

Mapping an Event Type to a BRM Service Type

You can map one event type to a single BRM service type (that is, one event type to a Pipeline Manager service code). For example, to rate GPRS and GSM usage events, you map the services as shown in [Table 18–2](#):

Table 18–2 Mapping an Event Type to a Service Type

Service	Event
/service/telco/gsm/telephony	/event/delayed/session/gprs

Table 18–2 (Cont.) Mapping an Event Type to a Service Type

Service	Event
/service/ip/gprs	/event/delayed/session/telco/gsm

You enter service-to-event mappings in Pricing Center.

Mapping Multiple Event Types to a BRM Service Type

To rate multiple events for a service, you map multiple event types to a single service type. For example, to rate GSM and GSMTEL usage events for GSM Telephony service, you map the `/event/delayed/session/telco/gsm` and `/event/delayed/session/telco/gsmstel` event types to the `/service/telco/gsm/telephony` BRM service type.

Rating multiple events for a service involves the following:

1. [Enabling the BRM Database to Store Multiple Event Types per Service](#)
2. [Populating the Event Types in the EDR Container](#)
3. [Configuring Output Streams Based on Service Code and Event Types for iRules](#)

Enabling the BRM Database to Store Multiple Event Types per Service

To map multiple event types to a service type, you must update the one-to-one event type and service type mapping restriction in the BRM database. To do so, update the `IFW_REF_MAP` table.

By default, the `IFW_REF_MAP` table holds a `PK_IFW_REM` primary key constraint on the `ID` and the `REF_OBJ` columns. To update the one-to-one event type and service type mapping restriction in the BRM database, hold (or place) the `PK_IFW_REM` primary key constraint on the `ID`, `REF_OBJ`, and `REF_PARAM` columns.

For information about the fields in the `IFW_REF_MAP` database table, see the documentation in *Pipeline_home/database*.

Populating the Event Types in the EDR Container

When multiple event types are mapped to a service type, you must ensure that each event type is mapped to the appropriate service code.

You create an iScript or an iRule that will determine the correct event type for a service type in an EDR and populate the `DETAIL.EVENT_TYPE` field with the event type *before* passing the EDR through the following modules:

- `FCT_Account` module.
- `FCT_AccountRouter` module for the account router instance of the Pipeline Manager.

For more information, see the discussion on creating iScripts and iRules in *BRM Developer's Guide*.

Configuring Output Streams Based on Service Code and Event Types for iRules

By default, the `IRL_EventTypeSplitting` iRule sends EDRs to separate output streams based on the service codes. When you configure multiple event types for a service type, you also map a service code to multiple event types. Therefore, you must update the `IRL_EventTypeSplitting` iRule and the data file to use the event types that are

populated in the `DETAIL.EVENT_TYPE` field along with the service code to send EDRs to separate output streams.

To configure the output streams based on service codes and event types:

1. Update the `IRL_EventTypeSplitting` iRule to add a condition for checking the event types.

The following `IRL_EventTypeSplitting` iRule sample shows the additional condition to check the event types:

```
CONDITION:
edrString(DETAIL.INTERN_SERVICE_CODE) =~ "${1}";
edrString(DETAIL.EVENT_TYPE) =~ "${2}";
```

2. Edit the `IRL_EventTypeSplitting.data` file to include the event types. For example, the following entries map the GSM service code to the `TELOutput` and `SMSOutput` streams:

```
GSM;/event/delayed/telco/gsm/telephony;TELOutput
GSM;/event/delayed/telco/gsm/sms;SMSOutput
```

Example Procedure to Map Multiple Event Types to a Service Type

The following procedure is an example of how you map the `/event/delayed/session/telco/gsm/telephony` and `/event/delayed/session/telco/gsm/sms` event types to a single `/service/telco/gsm` service type.

1. Update the `IFW_REF_MAP` database table to hold (or place) the `PK_IFW_REM` primary key constraint on the `ID`, `REF_OBJ`, and `REF_PARAM` columns.
2. This step is optional. In Pricing Center, map the `/service/telco/gsm` service type to two internal service codes, `GSM1` and `GSM2`. See the discussion on managing services in the Pricing Center Help.
3. Map the `/event/delayed/session/telco/gsm/telephony` and the `/event/delayed/session/telco/gsm/sms` event types to the `/service/telco/gsm` service type. See "[Mapping Event Types to Services](#)".
4. Update `/config/event_map` to map the `/event/delayed/session/telco/gsm/telephony` and the `/event/delayed/session/telco/gsm/sms` event types to the `/service/telco/gsm` service type.
5. Create a product with a `/service/telco/gsm` service type and configure the following two event types:
 - `/event/delayed/session/telco/gsm/telephony`
 - `/event/delayed/session/telco/gsm/sms`
6. Map the `/event/delayed/session/telco/gsm/telephony` and the `/event/delayed/session/telco/gsm/sms` event types to the required pipeline rate plan.
7. Create an iScript called `ISC_EventType.isc` to contain the logic that populates the `DETAIL.EVENT_TYPE` field with the appropriate event type. See the discussion on creating iScripts and iRules in *BRM Developer's Guide*.
8. Update the `IRL_EventTypeSplitting` iRule to map each combination of service code and event type to an output stream as follows:

```
GSM;/event/delayed/telco/gsm/telephony;TELOutput
GSM;/event/delayed/telco/gsm/sms;SMSOutput
```

```
GSM1 ; /event/delayed/telco/gsm/telephony;TEL1Output  
GSM1 ; /event/delayed/telco/gsm/sms;SMS1Output
```

Mapping Usage Classes

To create usage class mappings, you use Pricing Center to set up the data that specifies how to map the external data to the internal usage class. For example, you can specify that if the external supplementary service code for call forwarding is **41**, the internal usage code is **CW**. In addition, you can use various EDR attributes to create variations on supplementary service for rating purposes. For example:

- Service code mapping. Use the usage class to create special virtual services and products; for example, you can use the combination of telephony and call forwarding to create value-added services.
- ServiceClass-Mapping: Use the usage class to create special sub-services or quality of services (for example, service-level agreements).
- UsageScenario-Mapping: Use the usage class to map to impact categories.
- RateAdjustment: Use the usage class to provide discounts or adjustments in individual EDRs.

The mappings are based on the following data:

- External usage class
- Wholesale zone
- Tariff class and subclass
- Record type
- Connect type and Connect subtype, for example, anonymous, from another network, or direct
- Transit area code
- APN address when you use GPRS or UMTS technology
- SS (supplementary service) packet

The mapping is performed by the FCT_UsageClassMap module. You can configure this module to use a specific map group.

About Setting Up Usage Class Mapping

To set up usage class mapping:

1. Create internal usage classes in Pricing Center.
2. Map the external data to the internal usage classes in Pricing Center.
3. Configure the FCT_UsageClassMap module.

This module maps usage classes by using the data you entered in Pricing Center.

Mapping Usage Types

A usage type represents an attribute of a customer's account that is used in rating but which is not available from the network (for example, closed usergroup calls, friends-and-family calls, VIP calls, and birthday calls). A usage type is dynamically determined by comparison patterns using the A number and the B number.

You define usage types to represent the following customer information in an EDR:

- An ERA, such as an ERA for Friends and Family calls.
The usage type used for rating ERAs is determined by the call origin and destination data in the EDR. For example, if the destination number in an EDR matches the number on the account's friends-and-family list, the corresponding friends-and-family usage type is added to the EDR and is used during rating.
For information about ERAs, see "About Extended Rating Attributes for Telco Services" in *BRM Telco Integration*.
- A group to which a customer belongs.
This usage type is used by filter sets to apply system products and system discounts to a select group of customers. For example, if the EDR's `DETAIL.CUST_A.CUST_SEG_LIST` field matches one or more values you define, the corresponding usage type is added to the EDR.

Note: You must create your own application to add data to the `CUST_SEG_LIST` field.

For information about filter sets, see "About Using Filter Sets to Apply System Products and Discounts" in *BRM Configuring Pipeline Rating and Discounting*.

The usage types that you define are available to all rating scenarios.

Usage types are added to EDRs by running the `IRL_UsageType` iRule. This iRule performs no rating, it just adds the usage type to the EDR `DETAIL` block. You specify the rules for mapping EDR data to usage types in the files used by the `IRL_UsageType` iRule.

About Setting Up Usage Type Mapping

To set up usage type mapping:

1. Configure the `IRL_UsageType` iRule files. See the following sections:
 - [Configuring the `IRL_UsageType` iRule for ERAs](#)
 - [Configuring the `IRL_UsageType` iRule for Filter Sets](#)
 - [Configuring the `IRL_UsageType` iRule for Both ERAs and Filter Sets](#)
2. Configure the `FCT_IRule` module. See "[Configuring the `FCT_IRule` Module to Run the `IRL_UsageType` iRule](#)".
3. Create usage types in Pricing Center.

Configuring the `IRL_UsageType` iRule for ERAs

The `IRL_UsageType` iRule uses the following files in the `Pipeline_home/iScriptLib` directory. The directory contains sample files.

- **ISC_UsageType.isc.** This file contains the iScript code that determines whether an EDR qualifies for a usage type. This script reads EDR fields and sets a series of variables. It then assigns a usage type to the EDR based on a mapping of variables to usage type.

You can modify the `ISC_UsageType.isc` file to customize how to enter ERA data in Customer Center. For example, the following line uses a date format of `YYYY.MM.DD`:

```
String strCallCombineYearMonth = "." + strCallMonth + "." + strCallDay;
```

You can change this iScript to support a date format of YYYY/MM/DD by doing the following:

```
String strCallCombineYearMonth = "/" + strCallMonth + "/" + strCallDay;
```

- **IRL_UsageType.irl.** This file contains the rules for mapping the variables to values. If the EDR contains an attribute that matches a variable, the value for that variable is set to **Yes**. Otherwise it is set to **No**.
- **IRL_UsageType.data.** This file contains the rules for assigning a usage type to the EDR. If the values of the variables match those specified in this file, the corresponding usage type is set in the EDR.

Configuration example

Note: The sample configuration shows how to configure the iRule for ERAs only. It does not show how to configure the iRule for filter sets, which is done differently. For information, see ["Configuring the IRL_UsageType iRule for Filter Sets"](#) and ["Configuring the IRL_UsageType iRule for Both ERAs and Filter Sets"](#).

ISC_UsageType.isc

For ERAs, this file needs determination logic as shown in the following examples for the Friends and Family and Customer to Customer ERAs, respectively. This is in addition to the included **ISC_UsageType Function.isc** file, which is executed when a condition is fulfilled.

```
isFriendsAndFamily = compareProductERAforProfileAttributes
(profileNameFriendsFamily, "NUMBER",
edrString(DETAIL.B_NUMBER), serviceType, EXACT_MATCH);
```

```
if (edrNumDatablocks( DETAIL.CUST_A) > 0 and
edrNumDatablocks(DETAIL.CUST_B) > 0)
isCustomerToCustomer = "Y";
```

IRL_UsageType.irl

The following example shows the rules (conditions) for mapping variables to values for all ERAs. It also shows the result placeholder for the usage type. Each number corresponds to a position in the **IRL_UsageType.data** file.

```
CONDITION:
isRoaming =~ "${1}";
isInternational =~ "${2}";
isHomeRegion =~ "${3}";
isCustomerToCustomer =~ "${4}";
isSameClosedUserGroup =~ "${5}";
isSameCorporateAgreement =~ "${6}";
isSameCustomer =~ "${7}";
isSameSystemBrand =~ "${8}";
isSameSegment =~ "${9}";
isSameRateplan =~ "${10}";
isSameDiscountModel =~ "${11}";
isSameBillcycle =~ "${12}";
isBirthdayCall =~ "${13}";
isFriendsAndFamily =~ "${14}";
isHomeCell =~ "${15}";
```

```
RESULT:
edrString( DETAIL.USAGE_TYPE ) = "${16}";
```

IRL_UsageType.data

Each line in the **IRL_UsageType.data** file represents a different usage type mapping. Each position in a line corresponds to a variable (condition) in the **IRL_UsageType.irl** file. The last position specifies the usage type code to apply if the mapping matches. This position corresponds to the result variable in the **IRL_UsageType.irl** file, which in the example above is “\${16}”. Each position can have one of the following values:

- N = No
- Y = Yes
- . = any value

The following examples show the mapping and usage type for the Friends and Family and Customer to Customer ERAs, respectively.

```
N;N;.;.;.;.;.;.;.;.;.;.;.;.;Y;.;FNF
```

```
N;N;.;Y;.;.;.;.;.;.;.;.;.;.;.;.;CTC
```

Using the configuration examples above:

- If an EDR is not roaming or international (N in positions 1 and 2) and uses the Friends and Family ERA (Y in position 14), the usage type **FNF** is assigned.

This mapping is associated with these conditions in the **IRL_UsageType.irl** file:

```
isRoaming =~ "${1}";
isInternational =~ "${2}";
isFriendsAndFamily =~ "${14}";
```

- If an EDR is not roaming or international (N in positions 1 and 2) and uses the Customer to Customer ERA (Y in position 4), the usage type **CTC** is assigned.

This mapping is associated with these conditions in the **IRL_UsageType.irl** file:

```
isRoaming =~ "${1}";
isInternational =~ "${2}";
isCustomerToCustomer =~ "${4}";
```

Configuring the IRL_UsageType iRule for Filter Sets

The **IRL_UsageType** iRule uses the following files in the *Pipeline_home/iScriptLib* directory. There are samples of these files in that directory.

- **ISC_UsageType.isc**. This file contains the iScript code that determines whether an EDR qualifies for a usage type. This script reads EDR fields and sets a series of variables. It then assigns a usage type to the EDR based on a mapping of variables to usage type.
- **IRL_UsageType.irl**. This file contains a mapping to the contents of the **IRL_UsageType.data** file. For filter sets, this is a condition placeholder and a result placeholder.
- **IRL_UsageType.data**. This file contains one or more conditions and the usage type to be assigned. If the EDR contains attributes that match the conditions specified in this file, the corresponding usage type is set in the EDR.

Configuration example

Note: This sample configuration shows how to configure the iRule for filter sets only. It does not show how to configure the iRule for ERAs, which is done differently. For information, see "[Configuring the IRL_UsageType iRule for ERAs](#)" and "[Configuring the IRL_UsageType iRule for Both ERAs and Filter Sets](#)".

ISC_UsageType.isc

For filter sets, all you need is the included **ISC_UsageType Function.isc** file, which is executed when a condition is fulfilled.

IRL_UsageType.irl

The following example shows a placeholder condition that indicates the position in the **IRL_UsageType.data** file that contains the determination logic for applying the usage type. This example also shows a result placeholder for the usage type.

Note: Do not include a string in the condition to represent the EDR field being checked. For filter sets, the fields are identified in the **IRL_UsageType.data** file.

CONDITION:

```
"${1}";
```

RESULT:

```
edrString( DETAIL.USAGE_TYPE ) = "${2}";
```

IRL_UsageType.data

Each line in the **IRL_UsageType.data** file represents a different usage type mapping. Each position in a line corresponds to the positions in the **IRL_UsageType.irl** file.

The following examples show rules for assigning a usage type to the EDR. For filter sets, the rule contains both the determination logic and the usage type to apply. You can use Boolean operators to define complex rules.

```
isSegmentContains("1234") and isSegmentContains("GOOD");A1
```

```
isSegmentContains("789") or isSegmentContains("OK");B2
```

Using the configuration examples above:

- If the CUST_SET_LIST field is set to **1234** *and* **GOOD**, the usage type **A1** is assigned.
- If the CUST_SET_LIST field is set to **789** *or* **OK**, the usage type **B2** is assigned.

Configuring the IRL_UsageType iRule for Both ERAs and Filter Sets

You can configure one set of IRL_UsageType iRule files for both ERA and filter set usage type mapping. For background information, see "[Configuring the IRL_UsageType iRule for ERAs](#)" and "[Configuring the IRL_UsageType iRule for Filter Sets](#)".

Configuring the FCT_IRule Module to Run the IRL_UsageType iRule

The IRL_UsageType iRule is run by the FCT_IRules module. You must run this instance of the FCT_IRules module after the FCT_Account module and before the FCT_USC_Map module.

For information about the FCT_IRules module registry entries and EDR container fields, see "About Configuring iRules" in *BRM System Administrator's Guide*.

Adding Customer Balance Impact Data to EDRs

Note: Before you configure adding balance impacts to an EDR, you must configure internal service codes. See "[About Mapping Services](#)".

Before rating an EDR, you must add the customer's balance impact data to the EDR. This data includes the following:

- The POID of the service-level item that receives the balance impact.
- The start and end times of the accounting cycle that applies to the event. This data assures that the charges are applied to the correct bill item.

To find the balance impact data, you configure the following modules:

- Use the FCT_Account module to look up the balance impact data and add it to the EDR.
- Use the DAT_AccountBatch module to supply data to the FCT_Account module.

Configuring the DAT_AccountBatch Module

The DAT_AccountBatch module stores all customer data from the BRM database. The FCT_Account and FCT_AccountRouter modules use the data stored by the DAT_AccountBatch module.

Note: Because FCT_AccountRouter runs in a separate instance of Pipeline Manager, you configure separate instances of the DAT_AccountBatch module for the FCT_Account and FCT_AccountRouter modules.

The DAT_AccountBatch module stores the following types of data:

- *Dynamic data*, such as login names, products, accounts, and services. This data is updated by the DAT_Listener module. See "Configuring the DAT_Listener Module" in *BRM Installation Guide*.
- *Maintenance data*, such as products, plans, and deals. Maintenance data is not updated constantly. Instead, it is updated only when all data is reloaded into the database. See "Reloading Data into a Pipeline Manager Module" in *BRM System Administrator's Guide*.

To configure the DAT_AccountBatch module, see the following sections:

- [Specifying Not to Load Closed Accounts](#)
- [Specifying Whether to Load All Account Data](#)
- [Specifying How Much Account Data to Retrieve on Startup](#)

- [Configuring Product Validity Checking](#)
- [Configuring Account Product Validity Checking for Backdated Events](#)
- "Using Pipeline Manager with Multiple Database Schemas" in *BRM Configuring Pipeline Rating and Discounting*
- "Configuring the DAT_AccountBatch Module Database Connections and Threads" in *BRM System Administrator's Guide*

Specifying Not to Load Closed Accounts

When you start Pipeline Manager, DAT_AccountBatch loads all accounts into memory. This can affect start-up performance when there are a great number of accounts. To improve Pipeline Manager start-up performance, you can reduce the number of accounts loaded into memory by specifying not to load closed accounts.

To not load closed accounts, you set the **ClosedAccountDelay** registry entry in the DAT_AccountBatch registry. Specify the number of days before the current date for which closed accounts are not loaded:

ClosedAccountDelay = *number_of_days*

For example, if the system time is 11:00 a.m. on June 25 and the number of days is 10, accounts that were closed before 11:00 a.m. on June 15 are not loaded into memory.

The **ClosedAccountDelay** registry entry applies only to closed accounts. Accounts that are inactive, and open accounts with inactive services, are still loaded into memory.

If delayed EDRs arrive in the pipeline for closed accounts that were not loaded into memory, those EDRs are rejected and not rated.

The **ClosedAccountDelay** registry entry uses only the system time, not the virtual time. If you set the virtual time and restart the pipeline, the virtual time is not considered when loading closed accounts.

For example, on June 20 you set the virtual time to June 5 and create an account. You then change the virtual time to June 9, close the account, and set **ClosedAccountDelay** to 10 days. When you restart the pipeline, the account is not loaded because it was closed more than 10 days before the system time (June 20), even though the current virtual time is June 9. EDRs that arrive for that account are rejected.

Specifying Whether to Load All Account Data

Use the DAT_AccountBatch module **InitialLoading** registry entry to specify whether the initial loading of service and account data is performed. If the data is not loaded, loading occurs while processing. Login objects are always loaded.

The default is **True**. Setting this entry to **False** enables the system to start faster, but processing might be slower.

Specifying How Much Account Data to Retrieve on Startup

Use the DAT_AccountBatch module **RowFetchSize** registry entry to specify the number of rows of data to retrieve from the BRM database. The default is 1000.

The value you enter is used only during the initialization of DAT_AccountBatch to improve performance. After the DAT_AccountBatch startup is completed, **RowFetchSize** automatically resets itself to a more appropriate value of 50 to conserve memory.

Configuring Product Validity Checking

You can use the DAT_Account module **UseProductCreatedTime** registry entry to configure how product validity is checked:

- If you enable this entry, the product created time is not considered when establishing product validity. Only the start and end times are considered, so updates to products are valid immediately.
- If you disable this entry, the module checks the created time and the start and end time of a product when establishing the product's validity. In some situations this procedure can lead to unexpected results. The created time (PIN_FLD_CREATED_T) is system-generated whenever the product is modified. If the product is modified after its start time (PIN_FLD_START_T), there can be a short period when the product is invalid because its created time is later than the start time.

Configuring Account Product Validity Checking for Backdated Events

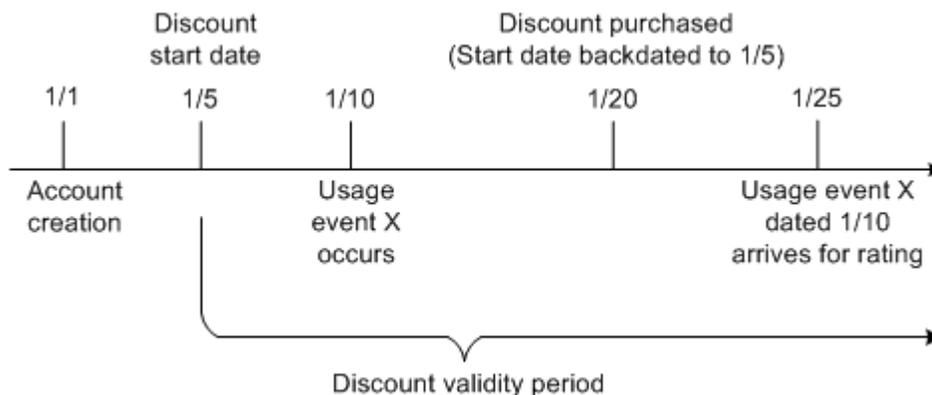
You can use the DAT_Account module **UseLatestProductAndDiscount** registry entry to configure validity checking for account products and discounts.

- If you enable this entry, when rating EDRs, DAT_AccountBatch retrieves account product and discount information based only on their start and end dates.
- If you disable this entry, DAT_AccountBatch retrieves account product and discount information based on their creation dates.

When product and discount information is retrieved during EDR rating, pipeline modules validate the account products and discounts by using the account creation date. This date is the time that the product and discount are purchased (the EFFECTIVE_T time in the AU_ACCOUNT_T audit table). If the product's validity start date is earlier than the actual purchase date, which can occur when purchase events are delayed, usage events might be incorrectly rated.

For example, an account is created on January 1. On January 20, the customer purchases a discount with a validity period that starts on January 5. On January 25, a delayed event that is dated January 10 arrives for rating. The discount module does not apply the discount because it uses the purchase date of January 20 instead of the discount's start date of January 5. Only events dated on or after January 20 are discounted. [Figure 18-1](#) shows a timeline for this example.

Figure 18-1 Product Validity for Backdated Events Example



If you set the **UseLatestProductAndDiscount** entry in the DAT_AccountBatch registry, account product and discount audit information is not retrieved. If you want delayed

events to use the account product and discount that was valid at the time the usage occurred, do not set this entry.

Getting Information about Loading Accounts

You can use DAT_AccountBatch semaphore commands to get data about the accounts loaded into the DAT_AccountBatch module.

- **PrintData** - Reports the account data for all accounts.
The data is written to the specified file. If you do not specify a file, it is written to **stdout**.
- **PrintDataLogin** - Reports the account data for a single account identified by the BRM login ID (usually the phone number).
The data is written to a file named *LoginID.lst*.
- **PrintDataSamples** - Reports the account data for a specified number of accounts, chosen randomly.
The data is written to a file named *Value.lst*. For example, if you enter **15**, the file is named **15.lst**.
If no value is provided, the module prints data for 10 accounts to **stdout**.

Configuring the FCT_Account Module

To find the balance impact data:

1. The FCT_Account module looks for the following data in the EDR:
 - The internal service code that indicates which data can be used to identify the account that generated the EDR. For example, if the internal service code is a telephony service, the identifying data is the A number. A different service might use the IMSI as the identifier.
You identify which data to use by using Pricing Center. See "[Specifying Which Data Is Used for Identifying Accounts](#)". Depending on the service, the module searches for either the login from a service object or an alias from a service object. An alias is typically a phone number or IMSI.
In rare cases, one customer's MSISDN can be the same as another customer's IMSI. You can configure account ID prefixes to use for handling duplicate telephone numbers. See "[Configuring Account ID Prefixes](#)".
 - The timestamp for the EDR. The timestamp is important because telephone numbers can be used by different accounts at different times.
2. The FCT_Account module uses the DAT_AccountBatch module to look up the account.

Note:

- If the A customer is not found, the EDR is rejected. If the B customer is missing, no error is generated.
 - Because phone numbers can be recycled, the search is made on data from BRM audit objects.
 - Accounts are loaded based on the service that is being rated by the pipeline. If an account does not own the service that the pipeline rates, it is not loaded.
-
-

3. The DAT_AccountBatch module returns the balance impact data.
4. The FCT_Account module inserts the customer data into the EDR.

Specifying Which Data Is Used for Identifying Accounts

Different services require different ways to look up accounts. For example, to look up a telephone account, you can use the A number to find the originating account and the B number to find the account that was called.

To specify which data to use for looking up an account, you use Pricing Center to map the service to the type of data. The data is stored in the IFW_ALIAS_MAP table.

You can set up multiple mappings to support different services (for example, specify different mappings for each pipeline).

Note: This data is used by both the FCT_Account module and the FCT_AccountRouter module.

Use the following data to specify how to identify accounts:

- The EDR container description that includes the field to use for identifying the account.
- The account reference. You must use one of the following:
 - Use **Account_CustA** for the A number.
 - Use **Account_CustB** for the B number.
- The internal service code (for example, **TEL** or **DATA**).
- The type (for example **Internal** or **Plugin**).
- The field ID (for example, **DETAIL.A_NUMBER**).

Note: Pricing Center shows the field name, but the database stores the field by using the field ID (for example, **20055**).

Configuring Account ID Prefixes

In some cases, the data used for identifying an account is not unique; for example, the MSISDN of one customer might have the same value as the IMSI of another customer. To avoid any conflicts, you can specify a prefix for the identifying data. This prefix data is stored in the IFW_REF_MAP database table.

You can specify a prefix based on the service type and event type, as shown in [Table 18-3](#). In this case, the number is prefixed with either **msisdn** or **imsi**.

Table 18-3 Prefix Configurations

Service	Event	Prefix	Example of Result
/service/telco/gsm/telephony	/event/delayed/session/gprs	msisdn	msisdn00491721234567
/service/ip/gprs	/event/delayed/session/telco/gsm	imsi	imsi00491721234567

To set up prefixes:

1. Customize the PCM_OP_NUM_POL_DEVICE_ASSOCIATE and PCM_OP_SIM_POL_DEVICE_ASSOCIATE policy opcodes to add a prefix to the MSISDN or IMSI.
2. In the Pipeline Manager IFW_REF_MAP object, use the IFW_REF_MAP.REF_COL attribute to specify the prefix, for example, **msisdn**.

Specifying Which Non-Currency Sub-Balances to Load on Startup

The DAT_BalanceBatch module uses the non-currency resource validity to select the non-currency sub-balances to load from the BRM database into pipeline memory. For example, if the Free Minutes resource validity is 30 days, at Pipeline Manager startup, DAT_BalanceBatch selects all Free Minutes sub-balances that were valid for the last 30 days.

You configure the non-currency resource validity in the resource configuration file (**pin_beid**). If the non-currency resource validity is not configured, DAT_BalanceBatch selects the sub-balances that were valid for 366 days by default

Important: When you set the non-currency resource validity, the sub-balances that do not fall within the validity range are not loaded into pipeline memory and therefore are not available for rating or rerating of call details records (CDRs). To make available the sub-balances that are required for rating and rerating, set the resource validity to an appropriate setting. For instance, if the CDRs are older than 366 days, you should set the validity greater than 366 days to ensure that all the sub-balances that are required are loaded and available for rating or rerating the CDRs. See "[Configuring the Non-Currency Resource Validity](#)".

If your business requires, you can configure DAT_BalanceBatch to load all the non-currency sub-balances available in the BRM database. To load all the non-currency sub-balances, use the **SelectiveSubBalLoad** entry in DAT_BalanceBatch registry. When this entry is set to **False**, DAT_BalanceBatch loads all non-currency sub-balances, including the sub-balances that are expired, into pipeline memory.

Important: When the BRM database contains a large number of non-currency sub-balances, loading all sub-balances leads to increased Pipeline Manager startup times.

Configuring the Non-Currency Resource Validity

To configure the non-currency resource validity:

Important: The **load_pin_beid** utility needs a configuration (**pin.conf**) file in the directory from which you run the utility. See "[load_pin_beid](#)" for more information.

1. Edit the **pin_beid** file in *BRM_Home/sys/data/pricing/example*.
Follow the instructions in the **pin_beid** file.

Note: If you are using an existing **pin_beid** file, you must manually add the **Validity_in_days** field to the syntax and set the resource validity for the non-currency resource. In this example, the resource validity for resource **1000010** is set to **30** days.

```
#beid r_mode round tol_min tol_max tol_pct beid_str_code symbol
event stage con_rule_id apply_mode Name Validity_in_days
#
1000010 0 1 0.000000 0.000000 0.000000 MIN Min * 0 0 1 Free
Domestic Minutes 30
```

Caution: The **load_pin_beid** utility overwrites the existing resources. If you are updating resources, you cannot load new resources only. You must load complete sets of resources each time you run the **load_pin_beid** utility.

2. Save and close the **pin_beid** file.
3. Run the following command:

```
load_pin_beid pin_beid
```

If you do not run the utility from the directory in which the **pin_beid** file is located, you must include the complete path to the file. For example:

```
load_pin_beid BRM_Home/sys/data/pricing/example/pin_beid
```

Tip: If you copy the **pin_beid** file to the directory from which you run the **load_pin_beid** utility, you do not have to specify the path or file name. The file must be named **pin_beid**.

See "[load_pin_beid](#)" for more information.

4. Stop and restart the Connection Manager (CM). See "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.

To verify that the non-currency resources were loaded, you can display the **/config/beid** object by using 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*.

5. Start Pipeline Manager.

At Pipeline Manager startup, DAT_BalanceBatch uses the non-currency resource validity in the **/config/beid** object to select the non-currency sub-balances to load into pipeline memory.

Modifying and Loading the EDR Container Description

By default, BRM supports the following EDR container description:

- **containerDesc.dsc:** The *rating EDR container description*. Defines the format into which CDRs are converted so that they can be processed for rating by a pipeline input module. See "BRM Rating EDR Container Description" in *BRM Configuring Pipeline Rating and Discounting*.

The EDR container description is in the *Pipeline_home/formatDesc/Portal* directory.

If the CDR requests you process include information that has no corresponding fields in the appropriate default EDR container description, you must modify the description to accommodate the custom data. See "[Modifying EDR Container Descriptions](#)".

After editing the rating EDR container description, you must load it into the appropriate Pipeline Manager database. See "[Loading EDR Container Descriptions](#)".

Modifying EDR Container Descriptions

If the CDR requests you process include information that has no corresponding fields in the default EDR container description, you must modify the description to accommodate the custom data.

The rating EDR container description (**containerDesc.dsc**) text file is located in *Pipeline_home/formatDesc/Portal*.

EDR Container Description Format

An EDR container description has the following format:

```
ComplexDataType
{
    String          StringVar; //description
    Integer         IntegerVar;
    Date           DateVar;
    Decimal        DecimalVar;
    ComplexDataType_2  complexVar;
}
ComplexDataType_2
{
    String          var1;
    Decimal         var2;
}
```

The following example shows how to format an EDR container description file:

```
// Comments
DETAIL // comment
{
String A_Number; // Comment will be filled into the ifw_edrc_field.description
Date STARTDATE;
Decimal duration;
CUSTOMER CUST_A; // New complex type must be defined later on
CUSTOMER CUST_B;
}
HEADER
{
// comment
String name;
Integer count;
}
CUSTOMER
{
String account;
String account_no;
Purchased_Product PRODUCT;
}
Purchased_Product
{
```

```
String name;  
Integer type;  
}
```

Note the following syntax and grammar rules:

- The file is case-sensitive.
- Real data types are the following:
 - String
 - Integer
 - Date
 - Decimal
 - Block (for example, `DETAIL.CUST_A`)
- The parsing is done in one step. Therefore, embedded complex data types must be defined later in the file.
- The name of each complex data type must be written in one line.
- The complex data type area must begin with a left brace (`{`) and end with a right brace (`}`). Each brace must be in its own line.
- Each data type variable pair must be written in one line. The line must end with a semicolon (`;`).
- Comments must begin with double slashes (`//`) and are valid until the end of the line.
- Comments following a real data type line are copied to the `IFW_EDRC_FIELD.DESCRPTION` field.

Loading EDR Container Descriptions

After editing the rating EDR container description, you must load it into the Pipeline Manager database.

You use the **containerDescLoader.pl** utility to load an EDR container description into a database. (You can also use Pricing Center, but using this utility is faster.)

EDR container descriptions are stored in the following database tables:

- `IFW_EDRC_DESC`
- `IFW_EDRC_FIELD`

Note the following restrictions:

- The Container Description Loader utility can process only one EDR container description file at a time.
- You cannot update tables that already include data. You can initialize empty tables or delete the data from tables before adding new data. You cannot, however, delete existing data if there are references to it from elsewhere in the database.

Creating the EDR Load Utility Command File

The Container Description Loader utility uses a command file to load data into a database.

The syntax of the command file is as follows:

```

#comments
ECHO print this on default output
TYPE=Database type
SOURCE=Database source
USER=Database user
PASS=Database user password
INIT=TRUE | FALSE
EDRC_DESC=Pipeline name as specified in the registry
EDRC_DESC_FILE=Container description file
EDRC_DESCRIPTION=Description of pipeline name
RUN=Execute this command

```

This is a sample command file with only the required entries:

```

DB=PR01|zaphod|zaphod
EDRC_DESC_FILE= /FMD/Portal/containerDesc.dsc
EDRC_DESC='MY_IFW'
EDRC_DESCRIPTION='MY_IFW container description'

```

This sample includes optional entries:

```

##
## This is a sample command file for containerDescLoader.pl
##
##
## DB (database connect)
## database|user|password
##
DB=PR01|zaphod|zaphod
#
# initialization (deleting tables)
#
ECHO=#####
ECHO=# deleting of ifw_edrc_desc and ifw_edrc_field
ECHO=#####
#INIT=FALSE
INIT=TRUE
# container description file
EDRC_DESC_FILE= /FMD/Portal/containerDesc.dsc
# pipeline name as specified in the registry
# filled into IFW_EDRC_DESC.EDRC_DESC and IFW_EDRC_FIELD.EDRC_DESC
EDRC_DESC='MY_IFW'
# additional description
# filled into IFW_EDRC_DESC.NAME
EDRC_DESCRIPTION='MY_IFW container description'
# optional command processing
#RUN=prog

```

Before Running the EDR Load Utility

Before running the Container Description Loader utility, do the following:

- Ensure that the following components are installed:
 - Perl 5.004 or higher
 - CPAN generic database interfaces
 - **oraperl** Perl module
- If necessary, create the following database tables:
 - IFW_EDRC_DESC

- IFW_EDRC_FIELD
- If necessary, create the following sequence:
 - IFW_SEQ_FIELD_ID
- Use the `IFW_PERL_LIB` variable to find the required Perl modules.

Starting the EDR Load Utility

To start the Container Description Loader utility:

1. Go to the `TLS_ContainerDescLoader` directory.
2. Enter the following command:

```
containerDescLoader.pl [-c command_file].
```

Dropping or Upgrading Incomplete Calls after Changing the EDR Container Description

The `FCT_CallAssembling` module stores any incomplete call records while a pipeline is running. When you stop the pipeline and change its container description, these incomplete call records are then automatically in the wrong format for the pipeline to process. This section describes your two choices for processing these call records:

- Discard them. This is the fastest way of dealing with the problem, but you lose the revenue they represent. You can however, use these EDRs for auditing purposes. For the procedure, see "[Discarding Incomplete Calls after Changing the EDR Container Description](#)".
- Upgrade them to the new container description. You use the tools provided to create a data upgrade pipeline that reformats the EDRs into the new container description format. For the procedure, see "[Upgrading Incomplete Calls to the New Container Description](#)".

These sections apply to all pipelines using `FCT_CallAssembling` to assemble CDRs into EDRs for the pipeline to process.

Discarding Incomplete Calls after Changing the EDR Container Description

This section explains the steps necessary to delete your partially-assembled call records during the process of changing the EDR container description.

You use the `UpgradeFlushLimit` semaphore registry entry to `FCT_CallAssembling` to flush and discard any incomplete calls stored in a pipeline during an EDR container description change.

You then use the `EmitPartialEDROnUpgrade` startup registry entry to specify what happens to the EDRs flushed by the `UpgradeFlushLimit` semaphore:

- If the `EmitPartialEDROnUpgrade` registry entry is `False`, the EDRs are silently dropped and are never processed by the pipeline.
- If the `EmitPartialEDROnUpgrade` registry entry is `True`, partial EDRs are processed by the pipeline but are not rated. You can use these records for revenue assurance purposes, but they should be discarded to avoid recycling.

EDR Data Available for Auditing

By default, all flushed EDRs will contain the following data fields:

- CHAIN_REFERENCE
- LONG_DURATION_INDICATOR (set to P)
- NUMBER_OF_CDRS
- CHARGING_START_TIMESTAMP
- CHARGING_END_TIMESTAMP
- DURATION

The following additional fields will also be included if the FCT_CallAssembling module has **AssembleVolume** set to **True**:

- VOLUME_SENT
- VOLUME_RECEIVED
- NUMBER_OF_UNITS
- RETAIL_CHARGED_AMOUNT_VALUE
- WHOLESALE_CHARGED_AMOUNT_VALUE

All EDR fields listed in the **AssembledEDR** registry entry are included if the **L** segment has been received.

Upgrading Incomplete Calls to the New Container Description

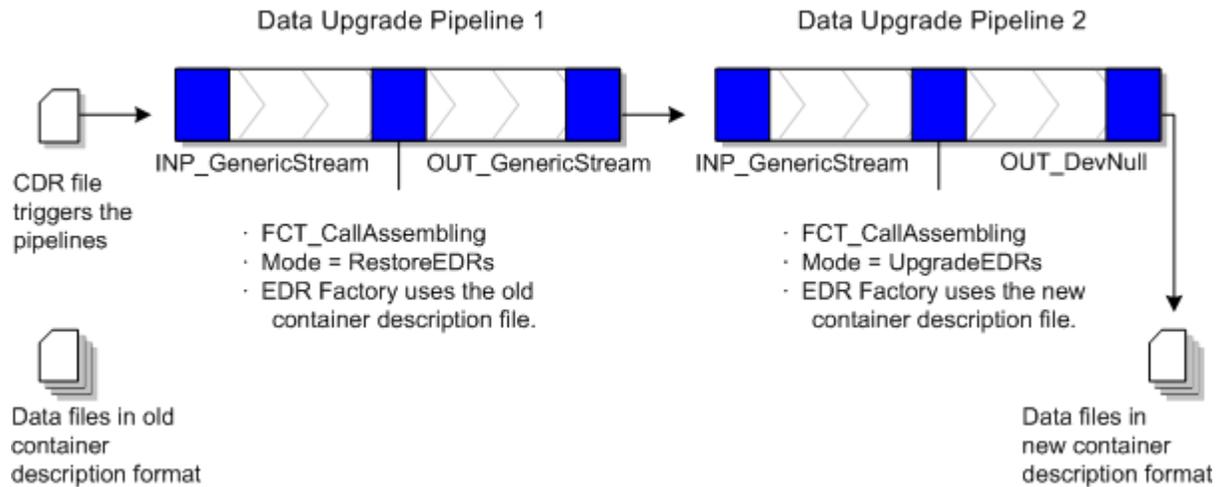
This section explains the steps necessary to update your partially-assembled call records to a new EDR container description. These tasks are necessary to correctly process any calls that are partially assembled at the time you shut down a pipeline to make the EDR container description changes.

Note: The tasks in this section apply only to BRM implementations that use FCT_CallAssembling. If you use an external mechanism to assemble wireless records, the instructions in "[Modifying and Loading the EDR Container Description](#)" are all you need to modify your EDR container description.

To correctly rate the incomplete calls with your new EDR container description, you must first update the data file maintained by FCT_CallAssembling. The following procedure explains how to do this by running the EDRs through two miniature pipelines that you will create. After this conversion, you can process these calls through the pipeline set up to process call records using the new container description file. Before you run the pipelines, you will also create new input and output mapping and grammar files and a new stream file.

[Figure 18–2](#) shows the process of upgrading stored EDRs to a new EDR container description.

Figure 18–2 Incomplete Call Description Update Process



Note: Using the **RestoreEDRs** and **UpgradeEDRs** modes, **FCT_CallAssembling** does not assemble calls. Instead, it only migrates them from one format to another.

Follow these steps to update EDRs being stored by **FCT_CallAssembling** to your new EDR format. This procedure assumes that you have already created your new container description file. For details see "[Modifying EDR Container Descriptions](#)".

Run this entire procedure for each EDR container being changed.

Tip: If you are upgrading multiple pipelines to the new container description, you only need to create one set of data upgrade pipelines. You can use them to upgrade all **FCT_CallAssembling** pipelines getting the new container description.

1. Run the **pin_container_to_stream_format** utility on your old EDR container description file.

For example:

```
pin_container_to_stream_format -c oldcontainer.dsc -g OLD_ -m OLD_ -s OLD_
```

This creates the stream, input and output grammar, and input and output mapping files that you will use to convert the partially assembled EDRs.

2. (Optional) Make any changes to the data fields in the new input grammar file to match your new EDR container description. The new container description file will have automatically supplied any necessary field *additions*. Edit the grammar file to specify any other changes to existing fields.
3. Create data upgrade pipeline 1. This pipeline will have only the following modules:
 - **INP_GenericStream** using your old input mapping file (actually any input mapping file can be used here).
 - **FCT_CallAssembling**. Set **Mode=RestoreEDRs** and set the **EdrFactory description** entry to your *old* container description file.

- OUT_GenericStream using the output mapping created by **pin_container_to_stream_format** in Step 1.
- 4. Create data upgrade pipeline 2. This pipeline will have only the following modules:
 - INP_GenericStream using the input mapping file created by **pin_container_to_stream_format** in Step 1 (modified as needed).
 - FCT_CallAssembling. Set **Mode=UpGradeEDRs** and set the **EdrFactory description** entry to your *new* container description file created in Step 1.
 - OUT_DevNull (removes the single CDR file you send in to initiate the pipelines).
- 5. Use a semaphore to stop the rating pipeline that is receiving the EDR container change. Wait for it to stop completely.
- 6. Back up the following files used by FCT_CallAssembling in your rating pipeline:
 - The most recent **.dat** file
 - All **.EDR** files
 - The most recent **.INDEX** file
- 7. Copy the most recent **.dat** file used by FCT_CallAssembling in your rating pipeline to the data directory used by FCT_CallAssembling in data upgrade pipeline 1 and data upgrade pipeline 2.
- 8. Copy the **.EDR** file used by FCT_CallAssembling from your rating pipeline to the EDR directory used by FCT_CallAssembling in data upgrade pipeline 1.
- 9. Copy the most recent **.INDEX** file used by FCT_CallAssembling in your rating pipeline to the index directory used by FCT_CallAssembling in data upgrade pipeline 1.
- 10. Use a semaphore to start the data upgrade pipelines.
- 11. Put one disposable CDR file in the input directory for data migration pipeline 1 to trigger the data upgrade. This file need only contain a single CDR, but it must use your old EDR container description format.

Note: This CDR is needed only to start EDR processing; it will get dropped during processing.

- 12. Ensure that data upgrade pipeline processes have finished. The time required for these pipelines to update all EDRs varies with the number EDRs being updated.
- 13. Use a semaphore to stop the data upgrade pipelines.
- 14. Move the **.dat** and **.EDR** files from the data upgrade pipeline 2 output directory to the input directory of the FCT_CallAssembling pipeline receiving the EDR container change.
- 15. Install the new EDR container description in your rating pipeline, and start the pipeline.

This completes the process of upgrading all EDRs stored in FCT_CallAssembling while upgrading your EDR container description. These incomplete calls are processed normally.

Rating by Date and Time with Pipeline Manager

This chapter describes how to set up time models for the Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager.

For information about Pipeline Manager, see "About Pipeline Rating" in *BRM Configuring Pipeline Rating and Discounting*.

About Rating by Date and Time with Pipeline Manager

A time model defines a combination of time periods. Each time period covers a specific time range for one or more days.

You use time models and time periods to charge different prices for the same service depending on the day and time.

Each time period contains the following information:

- **Day code:** The day or days the time period covers. For example, **Weekdays (Mon.-Fri.)**.
- **Time interval:** The time range the time period covers for the day code. For example, **08:00-17:00 Peak Time**.

Important: A time model must cover all the days of the week. Within a time model, although you can create multiple day codes, you cannot use the same day in multiple day codes.

For example, one time model contains these time periods and is associated with a price model that charges for GSM telephony service as shown in [Table 19-1](#):

Table 19-1 Rating by Date and Time

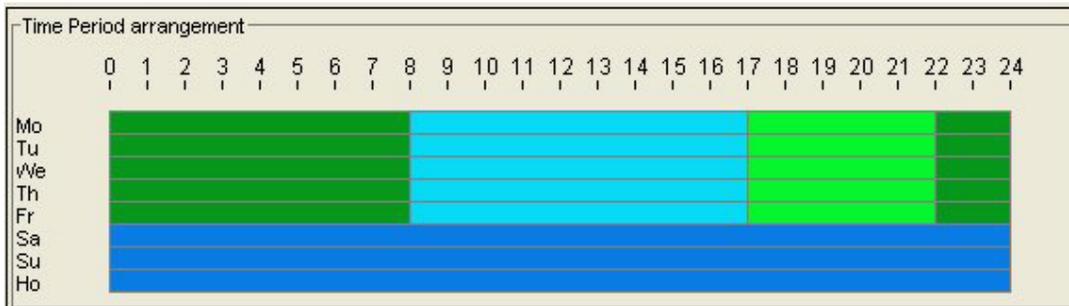
Time Period	Day Code and Time Interval	Charge from Price Model (Euro Cents per Minute)
WEEKPEAK - Weekdays Peak	Weekdays (Mon.-Fri.) 08:00 - 17:00	.15
WEEKOFF1 - Weekdays Off-Peak 1	Weekdays (Mon.-Fri.) 17:00 - 22:00	.12
WEEKOFF2 - Weekdays Off-Peak 2	Weekdays (Mon.-Fri.) 22:00 - 08:00	.10

Table 19–1 (Cont.) Rating by Date and Time

Time Period	Day Code and Time Interval	Charge from Price Model (Euro Cents per Minute)
WENDOFF - Weekend Off-Peak	Weekends (Sat., Sun., Holidays) 00:00-24:00	.09

Figure 19–1 shows a map from Pricing Center showing the time model in graphic form:

Figure 19–1 Example Time Model in Pricing Center



Note: Holidays are not treated as normal days, so you can add holidays to a time period without conflicting with any other days.

You include time models and time periods in rate plan configurations, which specify a price model for a rate plan version. When a call is rated, the time model and time period, with the service code and impact category, determine the charge.

See "[About Pipeline Rate Plan Configurations](#)".

About Holiday Calendars

A holiday calendar is a set of dates that you designate as holidays. Each date can either be a specific date valid only in one year, such as May 11, 2003 for Mother’s Day in the US, or a recurring date valid each year, such as January 1 for New Year’s Day.

You include a calendar in a rate plan. The rating module uses the calendar to determine if a date in an event detail record (EDR) is a holiday. You can include the same calendar in multiple rate plans, but you can also create different calendars for different rate plans.

Configuring Date and Time Rating

To configure date and time rating, do the following:

1. Create time models in Pricing Center and link them to rate plan configurations.
2. Create holiday calendars in Pricing Center and link them to rate plans. See "[Creating Holiday Calendars](#)".
3. Configure the DAT_TimeModel and DAT_Calendar modules.

Note: When you update time model and calendar data, you must use the **Reload** semaphore to reload data into the DAT_TimeModel and DAT_Calendar modules. See "Reloading Data into a Pipeline Manager Module" in *BRM System Administrator's Guide*.

Creating Holiday Calendars

To charge special rates for a holiday, do the following in Pricing Center:

1. Create a holiday calendar, including each holiday you want to include in a given rate plan.
2. Include the holiday calendar in a rate plan.
3. Define a day code that includes **Holiday**.

For example, the time period in [Figure 19–2](#) includes **Holiday** in its day code:

Figure 19–2 Defining a Day Code Including Holiday

The screenshot shows a form for defining a day code. The 'Code' field contains 'WEEKEND' and the 'Name' field contains 'Weekends (Sat-Sun+Hol)'. Below these fields is a 'Date Day' field which is empty. A list of days follows: Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday, and Holiday. Each day has a checkbox next to it. The checkboxes for Saturday, Sunday, and Holiday are checked. The 'Holiday' checkbox is circled in red.

4. Add the day code to a time period when defining a time model.
5. Include the time model in a rate plan configuration that is part of the same rate plan that includes the holiday calendar.

For more information, see Pricing Center Help.

About Special Day Rates

Use special day rates to create special rates for specific times or days, independent of time models. You can create two types of special day rates:

- **Global:** Special day rates that apply to all accounts in your system. See ["About Global Special Day Rates"](#) and ["Setting Up Global Special Day Rates"](#).

- **Account-specific:** Special day rates that apply only to one or more specific accounts in your system. You must create an extended rating attribute (ERA) for each account in Customer Center for this type of special day rate.

See "[Setting Up Account-Specific Special Day Rates](#)".

Important: When you create a special day discount, such as a birthday discount, any usage that begins on the discounted day and continues into the next, non-discounted day is fully discounted. Discounting does not split the charge packet at midnight but applies the discount for the entire call. For example, a customer receives a discount for 10% off on all birthday calls. On his birthday, the customer makes a 40 minute call that starts at 11:50 p.m. and ends on 12:30 a.m. the next day. The entire 40 minutes is discounted 10%.

About Global Special Day Rates

You can adjust charges for all accounts in your system based on the following:

- A specific date (for example, January 1, or a range of dates, for example April 1 through April 15)
- One or more days of the week (for example Sunday, or Tuesday and Thursday)
- A time period for the specified days
- A time period for all days

You can adjust charges in the following ways:

- A fixed amount (for example, reduce the charge by 1 dollar)
- A percentage discount (for example, reduce the charge by 10%)
- An absolute value (for example, replace the charge with 1 dollar)

This functionality is performed by the FCT_Dayrate module. To adjust the rate, the module finds EDRs that match the date and time, adjusts the charge, and overwrites the amount in the EDR.

For example, if you offer a 20% New Year's Day discount, the FCT_Dayrate module finds EDRs for calls made on January 1, calculates a 20% discount, and overwrites the charge in the EDR.

Special day charge adjustments are calculated after main rating. Therefore, rates based on time models are applied first, then adjusted separately by the special day rate.

Setting Up Global Special Day Rates

To set up special day rates that apply to all accounts, do the following:

1. Define special day rates in Pricing Center.
2. Configure the FCT_Dayrate module to calculate charges for date-based rates.
3. Configure the DAT_Dayrate module that stores the data used by the FCT_Dayrate module.

Note: When you update special day rate data, you must use the **Reload** semaphore to reload data into the DAT_Dayratemodule.

Setting Up Zones for Batch Pipeline Rating

This chapter describes how to set up zones for Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager.

For information about Pipeline Manager, see "About pipeline rating" in *BRM Configuring Pipeline Rating and Discounting*.

About Zoning

Zoning adds call data to EDRs that can be used for rating. A *zone* is a combination of origin and destination area codes. You create zones to rate calls based on their origin and destination. You can also define zones for different services. For example, you can create zones for rating:

- Local calls.
- National calls.
- International calls.
- Wholesale rating of roaming usage.
- National calls for specific services such as SMS.

There are two types of zones:

- *Standard zones*. These are determined by evaluating the area code of the calling number (the A number) and the called number (the B number). See "[About Standard Zones](#)".
- *Geographical zones*. These are determined by evaluating the distance between the origin and destination of the calls. See "[About Geographical Zones](#)".

All zones are mapped to impact categories. You combine impact categories with service usage data to create the final impact categories used by Pipeline Manager as the primary rating attribute. For more information, see "[About Impact Categories](#)".

When you configure zones in Pricing Center, you configure the following:

- Standard zones.
- Geographical zones.
- APN maps (for rating GPRS events).
- Zone models, which are collections of zones.
- Impact categories.

About Impact Categories

You use Pipeline Manager impact categories to rate calls based on event attributes such as call origin and destination, and service attributes such as call forwarding. For example, you can create impact categories to rate:

- Local calls.
- Long distance calls.
- Local call forwarding.
- Long distance call forwarding.
- Local mailbox inquiries.
- National mailbox inquiries.

You create an impact category for each zone and usage scenario that you define. A usage scenario describes the type of call, for example, a long-distance mailbox inquiry. You map the following data to impact categories:

- In zone mapping, a zone and service code. The impact category and zone name are the same.
- In usage scenario mapping, a service code, service class, usage class, usage type, and zone.

You must define all possible impact categories before you can use them in a zone model. Before defining impact categories, it is helpful to know the following:

- How many and what kinds of zones you must configure, for example:
 - Domestic zones and international zones.
 - Zones for rating all wireless usage types or for specific usage types only, such as telephony, fax, and SMS.
 - Zones for rating retail usage and roaming usage.
 - No-charge zones such as toll-free and emergency destinations.
- How many and what kind of usage scenarios you must configure, for example:
 - Mailbox inquiries
 - Conference calling
 - Call forwarding
 - Family and friends calls

Your impact category names should reflect the usage scenario. For example, if you offer service-level agreements, you might create three impact categories called **Bronze**, **Silver**, and **Gold**.

For more information on creating usage scenarios, see "[Setting Up Usage Scenario Mapping](#)".

About Wholesale and Retail Impact Categories

An impact category can be used for either wholesale rating or retail rating. *Retail rating* applies to usage by your subscribers on your network. *Wholesale rating* applies to usage by visiting subscribers, for example, roaming calls.

You define impact categories according to your retail and wholesale needs. However, when you create impact categories, you do not specify whether they are used for wholesale or retail rating. Instead, you enter the impact category name in the

wholesale and retail impact categories or zone fields when configuring zones and usage scenarios. You must enter values for both wholesale and retail impact categories and zones. When calls are rated, the type of impact category used is determined by whether the rate plan model type is wholesale or retail.

About Standard Zones

A *standard zone* is a combination of an origin and destination area code and one or more services such as telephony and SMS. You assign an impact category to each zone that you define. For example, in [Table 20-1](#), telephony origin and destination area codes are mapped to one of three impact categories; **Local**, **Toll**, and **LongDistance**:

Table 20-1 Impact Categories and Destination Area Codes

Origin Area Codes	Destination Area Codes	Destination Area Codes	Destination Area Codes
	0123	0234	0345
0123	Local	Toll	LongDistance
0234	Toll	Local	LongDistance
0345	Toll	LongDistance	Local
0456	LongDistance	LongDistance	Toll

About Geographical Zones

A *geographical zone* is the distance between the origin and destination of a call. You set up geographical zones to rate calls by distance. This is useful when:

- Customers are located very near the border between two area codes. For example, if a customer in one area code calls a person two blocks away in another area code, you do not want to charge for a long-distance call.
- The distance covered by an area code is very large and you want to use several rates within the same area code. You do this by mapping sets of latitude and longitude coordinates to the same area code in a geographical model.

To set up geographical zones:

- Define geographical zones and map them to distances.
- Create geographical models to:
 - Define the area codes that make up each zone.
 - Map latitude and longitude coordinates to area codes.

Geographical mapping uses the area codes and geographic coordinates to compute the distance and assigns the zone that matches that distance to the call record.

Displaying Zoning Information on an Invoice

You can enter a zone description that can be included in invoices. To do so, in Pricing Center, enter the zone description in the **Description** field in the following dialog boxes:

- Standard Zone
- Geographical Zone
- Usage Scenario Mapping

You can use a maximum of 2,000 characters in the description.

Important: To include zone descriptions on your invoices, you must configure the DAT_Zone and the FCT_USC_Map modules to load invoice description data into memory.

About Pipeline Manager Zone Modules

Zoning is performed by the following function and data modules:

- The FCT_APN_Map module adds Access Point Name (APN) data for the following:
 - To define zones.
 - To adjust impact categories.See "[Setting Up APN Mapping](#)".
- The FCT_PreRating module calculates zones and creates impact categories. This is the primary zoning module. See "[Setting Up Prerating](#)".
- The FCT_USC_Map module adjusts zones based on service attributes. See "[Setting Up Usage Scenario Mapping](#)".
- The DAT_Zone module handles zoning data for the zoning function modules.

The following function and iScript modules are used for optional zoning features:

- The FCT_Zone module calculates zones when you run Pipeline Manager for real-time zoning.
- The FCT_SegZoneNoCust maps zone models to segments. Use segment rating for collecting business information, for example, comparing the charges for the same events by using different rate plans.
- The FCT_MainZoning performs zoning for segment rating. It is used when a pipeline is used only for zoning.
- The ISC_SetSvcCodeRTZoning iScript updates DETAIL.INTERN_SERVICE_CODE EDR field with the customized service code value when different service codes are mapped to the same service type in the pipeline database and a specific service code is used in the zoning table.

About Zone Models

Every zone and usage scenario is mapped to an impact category. You create zone models to define the set of zones and impact categories available in the rate plan. For example, if you have a wireless rate plan that includes mailbox inquiries and SMS messaging, your zone model might include the following:

- Zones that map in-network area codes.
- Zones that map in-network area codes to area codes in your roaming partner networks.
- Usage scenario maps for mailbox inquiries.
- Usage scenario maps for SMS usage.

Your zone models can be simple or complex, depending on your business needs. For example, you might create just one zone model that includes all possible usage scenarios or many zone models that group more specific usage scenarios.

About Hierarchical Zone Models

You can also optionally create hierarchical zone models by linking them. Hierarchical zone models are useful for customizing zones for particular customers. For example, if a company has an office overseas, you might want to use a special rate for calls between the home and remote offices while using the standard rate for all other within the same zone.

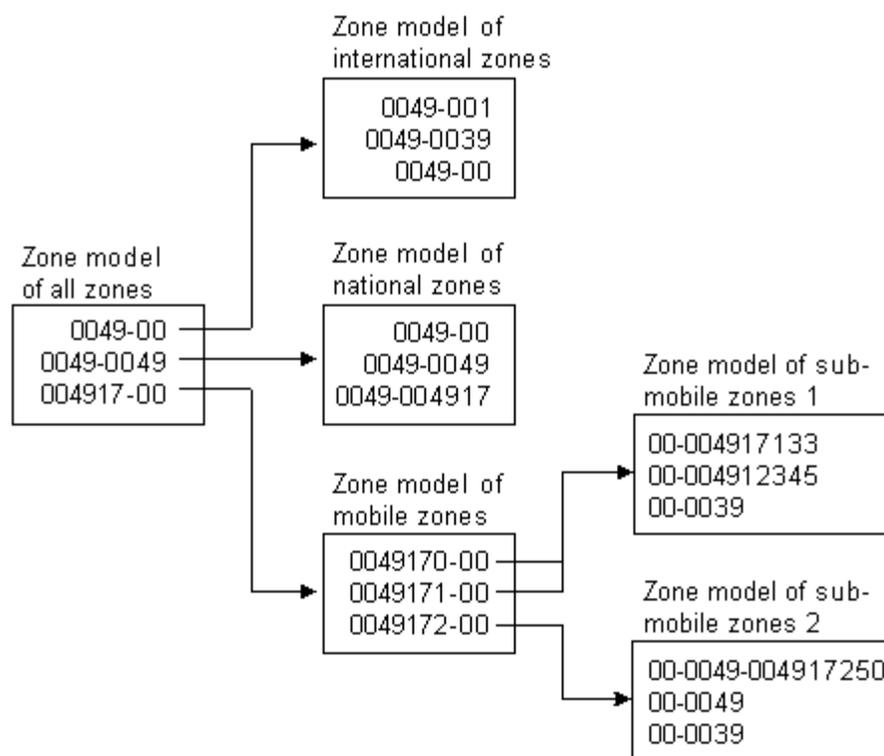
Hierarchical zone models also enable you to reuse zone models in different usage configurations, for example:

- You can include a zone model for rating Friends and Family calls in zone models for telephony usage.
- You can include a national zone model in an international zone model.
- You can include a geographical zone model inside a standard zone model to use several rates within the same area code. For more information, see ["About Geographical Zones"](#).

You can create as many levels of nested zone models as you like.

[Figure 20-1](#) shows a hierarchy of zone models for rating calls originating in Germany:

Figure 20-1 Zone Model Hierarchy for Calls Originating in Germany



Setting Up Zone Models

When you configure zone models in Pricing Center, you configure the following:

- Standard zones
- Geographical zones
- Usage scenario maps

To create zone models in Pricing Center, you complete these tasks:

1. Create impact categories for every zone and usage scenario you plan to define.
2. Create a zone model. You must create a zone model before you can create zones.
3. Create standard zones to map area code combinations.
4. If necessary, create geographical zones by performing these tasks:
 - Create geographical zones to map distance to impact categories.
 - Create geographical models to map area codes to longitude and latitude coordinates.
5. Create usage scenarios maps to map usage-specific data to impact categories.

For more information, see Pricing Center Help.

Setting Up APN Mapping

Use the FCT_APN_Map module to rate GPRS events. The module runs before zoning or after zoning:

- **Before zoning.** If you rate calls that access a GPRS data service, for example, to download content, there is no B number. In that case, the module runs before zoning. It maps the access point name (APN) to a packet data protocol (PDP) which serves as the B number that can be used for zoning.
- **After zoning.** If you rate GPRS calls made from one phone to another, for example, to exchange calendar appointments, the EDR includes the A number and B number. In this case, the zoning module has already created an impact category from the A number and B number. The FCT_APN_Map module uses the access point name (APN) to modify the wholesale and retail impact categories.

To configure APN mapping, you do the following:

1. Create APN maps. See "[About APN Maps](#)".
2. Configure the FCT_APN_Map module.

About APN Maps

To map APN numbers to PDPs before zoning or to impact categories after zoning, the FCT_APN_Map module reads data from the EDR and evaluates it according to the APN map. An APN map includes one or more mapping rules that specify the data that must be matched to apply the mapping.

You can create multiple sets of APN mapping rules, for example, different mappings for wholesale and retail rating.

You can use APN groups to assign a set of rules to a specific FCT_APN_Map module instance.

You can use the following EDR data to create an APN mapping rule:

- These values are compared with the EDR data when the module runs before or after zoning:
 - APN name (from the Associated GPRS Extension record APN_ADDRESS field)
 - Internal service code

- These values are compared with the EDR data only when the module runs after zoning:
 - Wholesale zone
 - Retail zone

You rank APN mapping rules when you create them. The first rule that matches the EDR data is used. The results of the mapping add the following data to the EDR:

- If the module runs before zoning, it adds the PDP address in the detail record INTERN_B_NUMBER_ZONE field.
- If the module runs after zoning, it adds the following:
 - Impact category
 - Wholesale zone
 - Resale zone

When you run the FCT_APN_Map module after zoning, different EDR fields are used depending on whether you use FCT_PreRate for zoning and rating, or FCT_Zone for zoning only:

- When FCT_PreRate is used (for zoning and rating):
 - The impact category is read from and written to the associated charge breakdown record.
 - The wholesale and retail zones are read from and written to the supplementary zone packet record.
- When FCT_Zone is used (for zoning only), the wholesale and retail zones are read from and written to the detail block of the EDR.

The following tables summarize the data mappings.

[Table 20–2](#) describes the EDR field mapping before zoning when there is no B number.

Table 20–2 Running FCT_APN_Map before Zoning

EDR Field	Compared against This Pipeline Manager Database Field
AssocCharge-APN group	IFW_APN_MAP.APN_GROUP
Detail-Internal Service Code	IFW_APN_MAP.SERVICECODE
Detail-Application	IFW_APN_MAP.ACCESSPOINTNAME

Running FCT_APN_Map after zoning:

For every zone breakdown record the following mapping is performed as shown in [Table 20–3](#):

Table 20–3 Running FCT_APN_Map after Zoning

EDR Field	Compared against This Database Field
AssocZone-APN group	IFW_APN_MAP.APN_GROUP
Detail-Internal Service Code	IFW_APN_MAP.SERVICECODE
Detail-Application	IFW_APN_MAP.ACCESSPOINTNAME
AssocZone-Wholesale Zone Result value	IFW_APN_MAP.ZONE_WS
AssocZone-Retail Zone Result value	IFW_APN_MAP.ZONE_RT

Creating APN Maps

Use Pricing Center to create APN maps and APN groups. APN groups are assigned differently depending on whether you run the FCT_APN_Map module before or after zoning:

- **Before zoning.** Specify the APN map in the **APNGroup** registry setting.
- **After zoning.** Specify the APN map in the zone model.

When you create mapping rules, you use regular expressions. See Pricing Center Help for more information.

Setting Up Prerating

Prerating is performed by the FCT_PreRating module. This module uses the data in the EDR to compute zones and assign an impact category to each charge packet.

The module works as follows:

1. It finds the rate plan to use by reading the rate plan code in the charge packet.

Note: When an EDR is associated with multiple rate plans, FCT_PreRating uses the highest priority rate plan. That is, it searches through all associated rate plans, from highest priority to lowest priority, until it finds a plan that matches the event's criteria. When two rate plans have matching priority, FCT_PreRating chooses the rate plan with the earliest start time.

2. In the rate plan, it finds the zone model.
3. It uses the zone model to calculate the zone and write the impact category to the charge packet. It reads the following information in the EDR:
 - Service code
 - A number
 - B number
 - Call start timestamp

The data used for determining zones and impact categories is handled by the DAT_Zone module. You configure the data by using Pricing Center.

To set up prerating, you do the following:

1. Create zone models and impact categories in Pricing Center.

You can also define zone data for the DAT_Zone module in ASCII files. See "[Creating Zone Data Files](#)".

Important: You can use DAT_Zone in file mode in an FCT_Zone/DAT_Zone combination only. Because file-based zoning cannot coexist with pre-rating, DAT_Zone in file mode does not work with FCT_PreRating.

2. Configure the FCT_PreRating module.
3. Configure the DAT_Zone module.

For information about configuring pipeline modules, see "Configuring Pipeline Rating" in *BRM Configuring Pipeline Rating and Discounting*.

Creating Zone Data Files

Zone data can be defined in ASCII files.

- Each line in a file contains one configuration entry.
- Fields are separated by semicolons (;).

Master Data File

Master data files defines the zone model codes in the format:

```
ZoneModelCode;ZoneModelType;GeoModel
```

where:

ZoneModelCode specifies a unique code for the zone model.

ZoneModelType specifies the zone model type; standard (S) or geographical (G).

GeoModel specifies the geographical link.

Master data file example:

```
ZM_ADD;S;100000
ZM_GEO;G;100000
```

Standard Zone File

Standard zone file defines standard zone models in the format:

```
ServiceCode;ValidFrom;ValidTo;Origin;OriginDesc;Destination;DestinationDesc;RtZone;WsZone;ZoneEntryName;ZoneDescription;AltZoneModel
```

where:

ServiceCode is the service code that applies to the zone.

ValidFrom is the date when the standard zone becomes valid, in the format DD.MM.YYYY. For example, **31.12.2004**.

ValidTo is the date when the standard zone becomes inactive. Use the same date format as above.

Origin is the origin area code for the zone.

OriginDesc is the description for the origin.

Destination is the destination area code for the zone.

DestinationDesc is the description for the destination.

RtZone is the impact category for retail rating to apply to the zone.

WsZone is the impact category for wholesale rating to apply to the zone.

ZoneEntryName is the name for the standard zone.

ZoneDescription is the description for the standard zone.

AltZoneModel is the alternate standard zone model that can be used.

Important: The *ZoneEntryName* and *ZoneDescription* fields must be included, even if they are not used.

Standard zone file example:

```
TEL;20030101;;0049;Germany;0049;Germany;NAT_MBI;NAT_MBI;;;
```

Geographical Zone File

Geographical zone file defines geographical zone models in the format:

```
Distance;ServiceCode;ValidFrom;ValidTo;WsZone;RtZone;ZoneEntryName;ZoneDescription;AltZoneModel
```

where:

Distance is the maximum distance for the geographical zone.

ServiceCode is the service code that applies to the zone.

ValidFrom is the date when the zone becomes valid, in the format DD.MM.YYYY. For example, **31.12.2004**.

ValidTo is the date when the zone becomes inactive. Use the same date format as above.

WsZone is the impact category for wholesale rating to apply to the zone.

RtZone is the impact category for retail rating to apply to the zone.

ZoneEntryName is the name for the geographical zone.

ZoneDescription is the description for the geographical zone.

AltZoneModel is the alternate geographical zone model that can be used.

Important: The *ZoneEntryName* and *ZoneDescription* fields must be included, even if they are not used.

Geographical zone file example:

```
500;*;19990101;;NAT_FIX;NAT_FIX;;;
```

Area Code Coordinate File

Area code coordinate file defines area codes used in a geographical zone in the format:

```
GeoModel;AreaCode;Longitude;Latitude;ValidFrom;ValidTo
```

where:

GeoModel is the description for the geographical link.

AreaCode is the area code to link.

Longitude is the longitude for the area code.

Latitude is the latitude for the area code.

ValidFrom is the date when the geographical link becomes valid, in the format DD.MM.YYYY. For example, **31.12.2004**

Setting Up Usage Scenario Mapping

To set up usage scenario mapping, do the following:

1. Create usage scenario maps. You have two options:
 - You can create usage scenario maps in Pricing Center. In this case, the usage scenario map data is stored in the Pipeline Manager database.
 - You can create a file that defines usage scenario maps. In this case, the FCT_USC_Map module reads the file. For information on creating the file, see "[Creating a Usage Scenario Map File](#)".
2. Configure the FCT_USC_Map module.

You specify the mode in which USC mapping is done.

- The **Rating** mode (the default) specifies that USC mapping is done using the zone model from the charge packets. Mapping in this mode provides the impact category for charge packets.
 - The **Zoning** mode specifies that USC mapping is done using the zone model from the EDR detail block. Mapping in this mode provides impact categories for the detail block.
3. Configure the DAT_USC_Map module.

When you use the database interface, you create USC map groups in Pricing Center, which contain the USC maps. You associate USC map groups with zone models. FCT_USC_Map uses the zone model ID in the EDR to identify the associated USC map group. The zone model ID is added to the EDR by the FCT_PreRating module. See "[Setting Up Prerating](#)".

Note: If there is no zone value in the EDR, FCT_USC_Map writes an empty string as the value of the zone to the output EDR. This should be considered when you set up the mapping.

Tip: You can use semaphore commands with the DAT_USC module to save USC Map data to a file.

- Use **PrintAllUscMapData** to save all of the USC Map data to a file.
 - Use **PrintUscMapDataForZoneModel** to save the data for a specified Zone Model ID.
 - Use **PreCompiledDataDir** to compile USC mapping data and save the data to a specified directory.
 - Use **NumberOfThreads** to specify the number of threads to use when compiling and loading the mapping data.
 - Use **UscGroups** to specify the USC groups for which to load rules.
-
-

About Usage Scenario Maps

Usage scenario mapping is performed by the FCT_USC_Map module. To define the final impact category, the module reads data from the EDR and evaluates it according to the usage scenario map. A usage scenario map includes one or more mapping rules that specify the data that must be matched to apply the impact category.

When you create usage scenario maps, you create a mapping rule for each impact category. You can also define the order in which the rules are evaluated. The impact category is derived from the first rule that matches the data. If no matching rule is found in the usage scenario mappings, then no mapping is performed.

When no usage scenario mappings are set up, FCT_USC_Map module uses the default USC map group. To use a default USC map group, you must create it and specify the default USC map group in the FCT_USC_Map registry.

You can use the following data to create a mapping rule:

- The EDR date and time. This data is read from the EDR **StartTimeStamp** field.
- Maximum event quantity, for example, 60 seconds. If the quantity exceeds this value, the map does not apply.
- Minimum and maximum wholesale amount.
- Usage class.
- Usage type.
- Service code.
- Service class.
- Wholesale and retail impact category.

When you create the mapping rules, you use regular expressions.

Creating a Usage Scenario Map File

The USC mapping rules are defined in an ASCII file.

- Each rule consists of a list of fields. The following table describes the meaning of each field.
- Every new line defines an adjustment rule.
- The position of each rule in the file determines its rank. The first matching rule is used.
- Fields are separated by semicolons (;).
- Comment lines start with #.
- Empty lines are allowed.

Important: The value of the field rank is ignored. The evaluation order of the rules is given by the order of the rules within the file.

Table 20–4 lists the fields in the file:

Table 20–4 *Fields in Usage Scenario Mapping File*

Position	Identifier	Description
1	USCGroup	Specifies the compare pattern for the source USC group.
2	Rank	This parameter is not evaluated. Instead, the rules are evaluated in the order they appear in the file.

Table 20–4 (Cont.) Fields in Usage Scenario Mapping File

Position	Identifier	Description
3	ValidFrom	<p>Specifies the start date for the USC mapping rule.</p> <p>This can either be a date with the format YYYYMMDD or a weekday with an optional timestamp, for example:</p> <ul style="list-style-type: none"> ■ 20030524 ■ SAT ■ MON16:00 <p>If the field is left empty, the earliest possible date 19010101 is used.</p>
4	ValidTo	<p>Specifies the end date for the USC mapping rule.</p> <p>This can either be a date with the format YYYYMMDD or a weekday with an optional timestamp.</p> <p>If the field is left empty, the latest possible date 20370205 is used.</p>
5	TimeFrom	<p>Specifies the beginning of the time period (in format HH:MM) the mapping entry is valid on each day of the validity period specified by ValidFrom and ValidTo. If the field is left empty, 00:00 is used.</p>
6	TimeTo	<p>Specifies the end of the time period (format HH:MM) the mapping entry is valid on each day of the validity period specified by ValidFrom and ValidTo. If the field is left empty, 24:00 is used.</p> <p>Example: To set up a mapping entry that is valid on weekends between 13:00 and 14:00, you must set ValidFrom=SAT, ValidTo=SUN, TimeFrom=13:00 and TimeTo=14:00.</p>
7	Quantity	<p>Specifies the maximum quantity value for an EDR container. If this maximum is exceeded, the mapping rule will not be used. If this field is left empty or a 0 is specified, the rule is valid for every quantity value.</p> <p>For example, to avoid mapping for calls longer than 60 seconds, set the value to 60.</p>
8	MinAocAccount	<p>Specifies the optional minimal wholesale amount value (the wholesale charge).</p> <p>The map is valid only if the EDR charge value is greater or equal to this setting.</p>
9	MaxAocAccount	<p>Specifies the optional maximum wholesale amount value (the wholesale charge).</p> <p>The map is valid only if the EDR charge value is less than or equal to this setting.</p>
10	SourceUsageClass	Specifies the compare pattern for the source usage class.
11	SourceUsageType	Specifies the compare pattern for the source usage type.
12	SourceServiceCode	Specifies the compare pattern for the source service code.
13	ServiceClass	Specifies the compare pattern for the source service class.
14	SourceImpactCategoryWS	Specifies the compare pattern for the source wholesale impact category
15	SourceImpactCategoryRT	Specifies the compare pattern for the source retail impact category.

Table 20–4 (Cont.) Fields in Usage Scenario Mapping File

Position	Identifier	Description
16	NewUsageType	Specifies the resulting usage type for this rule. If this field is left blank or set to 0, the usage type is not modified.
17	NewImpactCategoryWS	Specifies the compare pattern for the wholesale zone which should be mapped.
18	NewImpactCategoryRT	Specifies the compare pattern for the retail zone which should be mapped.
19	Description	Specifies the rule name. It is not used for rule evaluation.

Converting Zone Values for Output

Use an IRule script to convert the alphanumeric zone values (impact category name) into external result values before the EDR container is written to the output file. By default, the IMPACT_CATEGORY values from IFW_IMPACT_CAT are written to the output file as the zone value.

Setting Up Multi-Segment Zoning

Use multi-segment zoning to test different zoning models against the same EDR data. Use the FCT_SegZoneNoCust module and the FCT_MainZoning module to perform zoning for multi-segment zoning.

Use the FCT_SegZoneNoCust module to perform zoning when you cannot get customer account data about the EDRs. For example, you might want to use a test system that does not have access to account data. In that case, you configure the FCT_SegZoneNoCust module to find the segment based on the source network ID.

After the FCT_SegZoneNoCust module finds the segment, it adds the zone data for that segment to the EDR. It adds one associated zone breakdown record for each zone in the segment. The FCT_MainZoning module uses this data to perform the zoning and add the results to the zone packets.

To configure multi-segment zoning, configure the FCT_SegZoneNoCust and FCT_MainZoning modules.

Configuring Segments in the FCT_SegZoneNoCust Module

Use the FCT_SegZoneNoCust module **Segments** registry entry to specify the segment to use for each source network. Each rule defines the connection between the source network and the segment. For example:

```
26201 = SegmentD1
26202 = SegmentD2
```

Important: You cannot change these mappings during run time.

Part III

Using UE Loader to Import Service Usage Events

Part III describes how to load events into the Oracle Communications Billing and Revenue Management (BRM) database and rate them. It also includes information on troubleshooting this process.

Part III contains the following chapters:

- [About Rating Events Created by External Sources](#)
- [Loading Events from External Sources](#)

About Rating Events Created by External Sources

This chapter presents an overview of loading events from event log files into the Oracle Communications Billing and Revenue Management (BRM) database for rating.

Table 21–1 lists additional BRM documents that contain information about loading events from event log files:

Table 21–1 Links to Information on Loading Events from Event Log File

Documentation	Topic
About Importing Events from External Sources	Prerequisites for creating an event import template
Online help for the Universal Event (UE) Mapper application (start Developer Center and see UE Mapper online help)	Instructions for creating an event import template
Loading Events from External Sources	Instructions for loading events and handling errors
"Universal Event Loader" utility page	Syntax of utility you use to load events
"pin_uei_deploy" utility page in <i>BRM Developer's Guide</i> .	Syntax of utility you use to migrate event import templates from one database to another
"Creating custom bill items" in <i>BRM Configuring and Running Billing</i> .	Considerations for using custom bill items

About Importing and Rating Events

BRM can rate service usage in real time. To rate usage in real time, you need a BRM client application such as RADIUS Manager to capture billable events. For some services, it is easier to import and rate events recorded in event log files than to create a BRM client application that captures and rates the events in real time. You can import events from event log files that include data from Web servers and other sources.

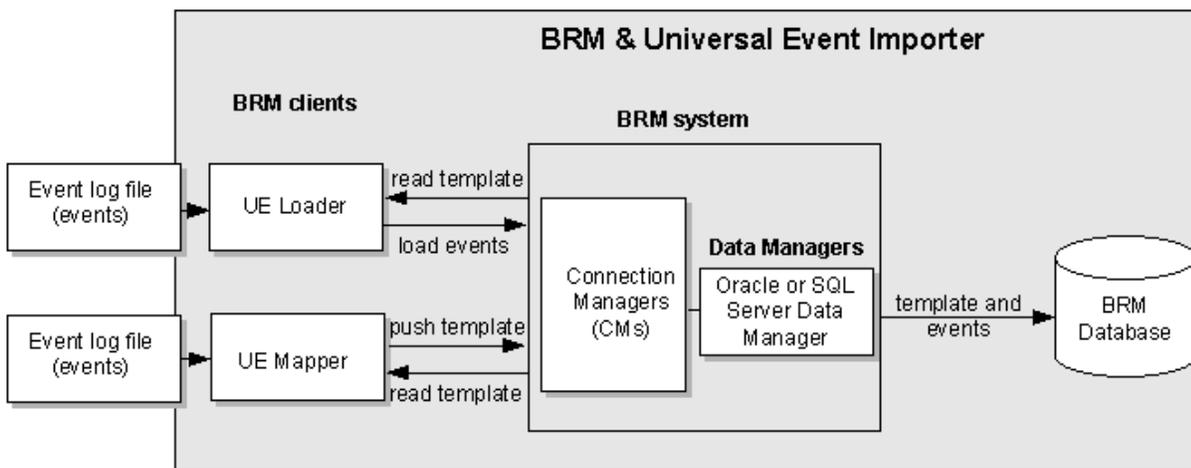
UE Importer consists of UE Loader, a command-line event loader that parses the event log file and loads the event into BRM, and UE Mapper, a client GUI application that you use to create event import templates that specify the file format. UE Loader uses the template to parse and load the event log file.

Figure 21–1 shows that:

1. UE Mapper:

- a. Reads the event log file when you create an event import template for the log file.
 - b. Saves the event import template to the BRM database.
 - c. Reads the template from the database when you modify the template.
2. **UE Loader:**
- a. Reads the event log file.
 - b. Uses the event import template for instructions on loading the events from the log file into the database.
 - c. Loads the events into the BRM database.

Figure 21-1 BRM and UE Importer



About Universal Event Mapper

To capture the events you want to import from event log files, use Universal Event (UE) Mapper to create an event import template. You can create multiple templates for your needs. For example, you might need one template for IP telephony data records and another for Internet dialup usage records. You can also create a single template to process both of these record types if they occur in the same event log file. Event import templates are stored in the BRM database in XML format.

For more information on creating event import templates, see "[About Importing Events from External Sources](#)". For step-by-step procedures on creating event import templates, see UE Mapper and see the Help.

About Universal Event Loader

To import and rate events from event log files, you use Universal Event (UE) Loader. UE Loader reads the event import template to load data from event log files into BRM as billable events. When the events are loaded, BRM rates the events and updates customer account balances. UE Loader is included in the BRM server software and is located in *BRM_Home/apps/uel*.

About Setting Up Event Importing

To import and rate events from event log files, you must do the following:

1. If you have not already done so, create a price list that includes the products and rate plans used for rating the events. You might need to define new event and service subclasses. For more information, see ["About Creating a Price List"](#).
2. Use UE Mapper to create a template that translates the event log file entries into BRM event fields. UE Loader uses the template when loading events. See ["About Importing Events from External Sources"](#).
3. Run UE Loader to import and rate the events. See ["Loading Events from External Sources"](#). You can load events manually or set up automatic loading by using Batch Controller. For information on automatic loading, see "Controlling batch operations" in *BRM System Administrator's Guide*.

You might need to create custom opcodes to import events if BRM opcodes cannot handle the type of data you must import. See ["About Creating Custom Opcodes"](#).

About Supported Event Log Files

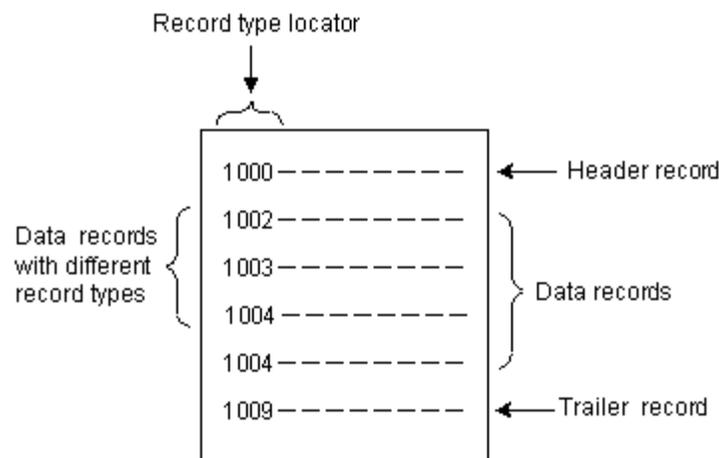
This section describes the data log files you can import (load event data) into the BRM database.

You can import ASCII or binary data from event log files. If your event log file is a binary hex file, you can import records that use a ["fixed width"](#) record format. If your event log file is a text file, you can import records that use a ["fixed width"](#), ["delimited"](#), or ["attribute value"](#) record format.

Event Log File Terminology

The event log file can contain a header record (optional), data records, and a trailer record (optional). The header record and the trailer record can be defined by a unique record type locator. The data records can be more than one record type as specified by their record type locators. [Figure 21-2](#) shows a sample event log containing the different record types.

Figure 21-2 Sample Event Log with Records



You specify whether your event log file has a header and trailer record when you create the event import template. If you do not specify that your event log file has a header and trailer record, UE Loader assumes all records in the event log file are data records.

Each of these three types of records (header, data, and trailer) might be identified by a unique record type. If you have a header and trailer record, and you do not define a record type for each, UE Loader assumes the first record that is imported is the header and the last record is the trailer.

Important: There can be only one header record and one trailer record in an event log file.

For more information on record types, see "[About Loading Event Records Based on Record Type](#)".

Supported File Types

You can import event log files with these file types:

text file type

Consists of data in ASCII text format. The record format for records of this file type must be delimited, attribute value, or fixed width.

binary hex file type

Consists of data as BCD (Binary-Coded Decimal) integers and EBCDIC (Extended Binary-Coded Decimal Interchange Code) strings. The record format for records of this file type must be fixed width.

Supported Record Formats

You can import the following record formats into BRM. All records in a log file must have the same record format.

attribute value

Fields within the records are specified by attribute-value pairs in this format: <name><attribute-value separator><value><delimiter>. For example, **login=bob**, is an attribute-value pair that uses an equal sign as the separator and a comma as the delimiter. Attribute-value pairs can occur in any order in the event log file. Records can be in one line or span multiple lines.

delimited

Records and record fields are delimited by ASCII characters. The delimiter consists of only one character, such as a comma.

fixed width

Fixed width data can be one of the following:

- **Continuous:** Each record has a fixed length
- **Delimited:** The fields have a fixed length

Within their specified length, fields can contain padding characters and can be left or right justified.

Note: If there are multiple record types, the records may or may not be uniform in length. Record structure can also vary between record types. For example, you can define columns 1 through 8 as the start time for IP telephony record types, and columns 1 through 5 as the start time for Internet dialup record types.

Supported Trailer and Header Records

Typically, there are three kinds of records in an event log file: header records, data records, and trailer records. A file can include one header record, several data records, and one trailer record. Data records can have any number of data record types, such as data records pertaining to email services, dialup IP services, IP telephony services, and so on. You specify record types in UE Mapper by specifying a record type locator, a field that identifies each record's type.

You specify that your event log file has a header and trailer record in the event import template so UE Loader does not load them into the database. The optional header and trailer records may or may not have a record type. If a header and trailer record is specified without a record type, the first record is treated as the header and the last record is treated as the trailer.

Note: Event log files are not required to have header and trailer records.

Header Records

There can be only one header record in a data log file. The header record must precede all data records and must use the same record format as all other records (delimited, fixed-width, or attribute-value format).

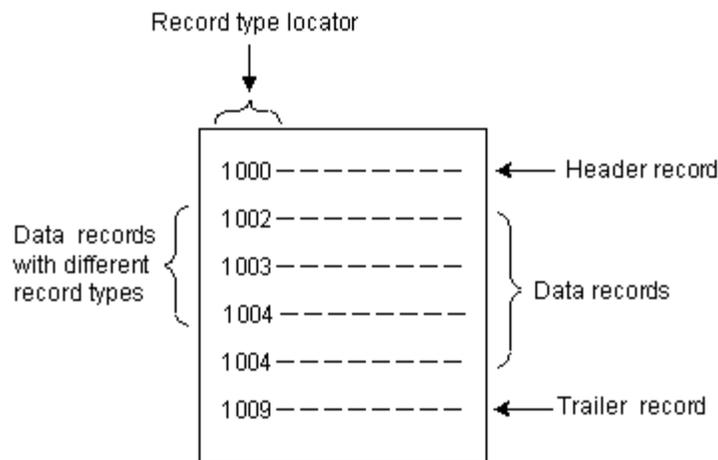
Header records usually contain information relevant to all events in the data records. A header record can also contain information that validates the status of the data log file, such as the number of expected data records in the file. You can set up a header check in UE Mapper to use this data to validate the number of records in the event log file.

Trailer Records

There can be only one trailer record in an event log file. It must use the same record format as all other records (delimited, fixed-width, or attribute-value format), and it must follow all data records.

Trailer records usually contain information to verify the status of the event log file. If your trailer records include this information, you can set up trailer checks by using UE Mapper to validate the event log file.

[Figure 21–3](#) shows an event log file with a header record, data records of several record types, and a trailer record:

Figure 21-3 Sample Event Log with Records

About Creating Custom Opcodes

If BRM opcodes cannot handle your event log files, you might need to create custom opcodes to load events.

About Custom Opcodes to Map Event Data to Accounts

You might need to create a custom opcode to find the customer accounts associated with the events in your event log files. Usually, the data in the event log file includes login names that existing BRM opcodes can use to find the account for each event. If the event log file does not include login names, you must write a custom opcode that finds the account based on other data in the event log file, such as the customer's IP address. See ["Creating Opcodes for Finding Accounts Associated with Event Data"](#).

About Custom Opcodes to Load Event Data

You might need to create a custom opcode to load the events into the BRM database. In many cases, you can use existing BRM opcodes to load event data, but to preprocess the data in the event log file either before rating or before storing the event data in the BRM database, you must create a custom opcode.

For example, if your event log files include a value for downloaded kilobytes, and you need the value represented in bytes, you can create an opcode to preprocess the data before loading the event. Also, if the event log files include area codes and phone numbers in different fields, you may want to concatenate the area code and the phone number in the opcode to store it as a single number in the database. For more information, see ["Creating Opcodes for Loading Event Data"](#).

About Loading Event Records Based on Record Type

Data records can have any number of data record types, such as data records pertaining to email services, dialup IP services, IP telephony services, and so on. You specify record types in UE Mapper by specifying a record type locator, a field that identifies each record's type.

One event log file can contain data records of several record types. When you create an event import template for log files with multiple record types, you specify the following items for each record type:

- The BRM event type associated with the data
- The opcode that loads the data
- The opcode that finds the account associated with the data
- The filters that are applied to load or log the data

About Ignoring Record Types When Loading

An event import template specifies which record types are loaded into your BRM database.

You might not want to load all event records from your event log files into BRM. For example, you specify a record type locator so you can use *filters* to prevent unnecessary record types from being loaded into the database. A *record type locator* is a unique field that identifies the different record types.

Using filters can improve loading performance. For example, in many IP telephony CDR files, there are **start accounting** event records and **start stop accounting** event records. The **start stop accounting** event records contain the necessary information to rate IP telephony calls, so it is not necessary to load the **start accounting** event records. Using filters to prevent loading **start accounting** records into the database significantly increases loading performance.

About Using Filters for Loading Events

You can set up filters in the event import template so that events in your event log file that satisfy specified criteria are handled as follows:

- Discarded (not loaded). See "[About Discarding Event Records Based on Record Content](#)".
- Logged to a separate file. You can log events that are loaded or discarded to a separate file. See "[About Logging Event Records Based on Record Content](#)".

Event filtering occurs after UE Loader does the following:

1. Reads the event log file.
2. Parses the contents of the event log file.

This task includes verifying whether each event in the log file is associated with a customer account in the BRM database. Thus, account verification is done for some events that might later be filtered out of the event log file.

3. Creates a cache file that contains the results of the parsing.

After creating the cache file, UE Loader applies any filters in the event import template to the events in the cache file, removing events that satisfy the filter-out criteria.

For instructions on setting up filters, start Developer Center and see the UE Mapper Help.

About Discarding Event Records Based on Record Content

Use filters to prevent loading certain event log file records into the database. Records that have particular data field values can be marked and rejected without causing errors in the loading process.

For example, if you have a list of accounts with an inactive account status, you can reject event records that have login fields associated with those accounts.

Another reason for rejecting records based on record content is when you have a minimum amount of usage for which you want to charge users. For example, you can reject records where the dialup session is less than fifteen seconds.

About Logging Event Records Based on Record Content

Use filters to log specific event log file records to a file that you then load into the database. The filter marks records that have particular data field values and logs them to a file.

For example, if you notice accounts with high volume usage, you might want to offer these customers a pricing plan for high volume users. A filter can mark and log the accounts that have a usage value above your filter criteria to create a list of customers to contact. You can also set up a filter to mark and log accounts that show signs of fraudulent activity.

You set up filters when you create the event import template. For instructions, start Developer Center and see the UE Mapper Help.

About Validating the Event Log File

If your event log file includes header and trailer records, you can set up checks to verify that the complete number or size of records is in the log file.

You set up header and trailer checks by using UE Mapper when you create the event import template. For instructions, start Developer Center and see the UE Mapper Help.

About Importing International Log Files

UE Loader supports log files in English, French, German, and Japanese. For instructions on setting up UE Loader for these languages, see "[Configuring UE Loader for International Log Files](#)". UE Loader supports UTF8 and Unicode encoding formats.

Importing and Rating Events from External Sources

You can rate service usage by importing usage events from log files, such as Web server log files and IP telephony call detail records (CDR) files, into BRM. This section provides a brief technical overview of how UE Loader imports event data from event log files into the BRM database for rating. It also explains how to prepare for creating an event import template.

For an overview of importing events, see "[About Rating Events Created by External Sources](#)". For instructions on loading data log file event data into the BRM database, see "[Loading Events from External Sources](#)".

About Importing Events from External Sources

To import usage events from log files, you create event import templates that specify the usage fields to import and how to import them. You use UE Mapper, a component of Developer Center, to create event import templates. After you create the templates, you use the UE Loader, a command-line utility installed with BRM, to test the templates and load events from your event log files into BRM. UE Loader reads the event import templates in the database for instructions on loading the events.

Overview of Preparing for Loading Events

There are seven steps to prepare for loading external event data into the BRM database:

1. If necessary, create an opcode to find the accounts associated with the event data to load. See ["Creating Opcodes for Finding Accounts Associated with Event Data"](#).
2. If necessary, create an opcode to load the event data. See ["Creating Opcodes for Loading Event Data"](#).
3. Configure UE Mapper to customize delimiter options. See ["Adding Options to UE Mapper Drop-Down Lists"](#).
4. If you use custom events and fields, understand the procedures for importing them. See ["Making Custom Fields Available for Mapping"](#) and ["Displaying the Description of Custom Events"](#).
5. Create an event import template using UE Mapper and save it to a test BRM database. See the UE Mapper online Help.

The event import template drives the event import process. It specifies:

- The event data to load from the event log file records.
 - The BRM storable object and storable object fields into which the event data is loaded.
 - The BRM opcode that finds the accounts associated with the event data (optional in some cases).
 - The BRM opcode that loads the event data.
6. Test the event import template by running UE Loader to ensure that event data can be loaded successfully from a sample event log file.
 7. Save the event import template to your BRM production database so that UE Loader can load external event data as needed.

How UE Loader Works

UE Loader first loads the event log file records into an object queue. It then reads data from the event import template to construct an flist with the account search criteria and pass the flist to an opcode that finds the account associated with the events. After UE Loader obtains this account, it reads data from the event import template to construct an flist with the event data and passes the flist to an opcode that loads the event data into the account. If UE Loader is set up to use multiple threads, it can load events into several accounts at once. For more information on multiple threads, see ["Setting the Number of Threads for Performance"](#).

Constructing an Flist by Using UE Mapper

For UE Loader to construct an flist, it reads the event import template for the flist instructions you specified. You can specify arrays in the template if the flist should include arrays: for example, to import multiple contacts. You use UE Mapper to populate the fields of an flist with the event data to import into BRM.

To see a valid input flist that was created to load event data from the log file into a specific BRM event object:

1. Load the sample event import template (*BRM_Home/apps/sample_handler/sample_template.xml*) into the BRM database:

```
pin_uei_deploy -t sampleTemplate -c -i sample_template.xml
```

2. Create a sample event log file named **SampleEvent** in a text editor and add the following contents:


```
Field1Field2Field3Field4Field5
v01/jun/2003:01:04:00 am PST01/jun/2003:01:07:00 am PST0812
v01/jun/2004:01:04:00 am PST01/jun/2004:01:07:00 am PST2013
v01/jun/2005:01:04:00 am PST01/jun/2005:01:07:00 am PST3015
Field1Field2Field3Field4Field5
```
3. Restart Developer Center.
4. Open UE Mapper in Developer Center.
5. Choose **Modify Template** from the **File** menu.
6. In the Modify Template dialog box, select **SampleTemplate** from the **Templates** list and click **Modify**.
7. In the Open Event Log File dialog box, specify the **SampleEvent** event log file and click **Open**.

Creating an Event Import Template

Before you can import events, you use UE Mapper to create an event import template. This section describes tasks you might need to do before you can create the template.

UE Mapper is a Developer Center application. UE Mapper includes detailed help for creating event import templates.

Before You Begin

To create an event import template:

- You should be familiar with the contents of the event log file you are importing, including:
 - What each record and field in the event log file means. Each record corresponds to an event in BRM, and each piece of data corresponds to a field in an event.
 - How the event log file is formatted, including which character signifies the end of a record and which character separates data fields in a record. For example, records might be separated by line ends, fields might be separated by commas, or records might contain continuous data in a fixed-width format.
- You must understand BRM events, in particular, which type of BRM events are associated with the log file events you are loading.
- You must understand BRM opcodes, including required field names and the data types of those fields.

Before you create an event import template, you might need to perform one or more of the following tasks:

- [Preparing Event Log File Data](#)
- [Creating Opcodes for Finding Accounts Associated with Event Data](#)
- [Creating Opcodes for Loading Event Data](#)
- [Making Custom Fields Available for Mapping](#)
- [Displaying the Description of Custom Events](#)

- [Adding Options to UE Mapper Drop-Down Lists](#)
- [Migrating Event Import Templates from One BRM Database to Another](#)

Preparing Event Log File Data

In some cases, you must preprocess the records in your event log file before you can use the file to create an event import template. You might need to inspect the data in the input records for errors and manipulate the event data before it can be imported into BRM.

Creating Opcodes for Finding Accounts Associated with Event Data

The BRM opcode commonly used to find customer accounts associated with event data is PCM_OP_ACT_FIND. This opcode uses the customer login name in the event record to search the account table in the database associated with the event. If your event records do not contain the customer login name, you must create a custom opcode that uses another field to find the accounts.

Important: The custom opcode must return the account Portal object ID (POID). It can also return the service, but it is not required to.

For example, if you have IP telephony call detail records (CDRs) that include the call origination number or call ID number of the user instead of the login name, you could create a custom opcode to map the call ID in the event to the account associated with that call ID.

Creating Opcodes for Loading Event Data

In some cases, you must create custom opcodes to load event data.

The BRM opcodes that are commonly used to load data from event log files into BRM are PCM_OP_ACT_USAGE and PCM_OP_ACT_LOAD_SESSION. If you use a method for rating events that is more complex than these opcodes can handle (the data you need to load is not in the format these opcodes expect), you cannot map data to them. You must create a custom opcode that can load your event data.

For example, when importing IP telephony call detail records (CDRs), you probably want to perform duplicate session checking and handle start and stop accounting events. PCM_OP_ACT_USAGE and PCM_OP_ACT_LOAD_SESSION do not handle that data, so you create custom opcodes to handle these transactions. For other examples for creating a custom opcode to load event data, see "[About Custom Opcodes to Load Event Data](#)".

Making Custom Fields Available for Mapping

If you create custom BRM fields, you must make these fields available to UE Mapper before they can be displayed for mapping. You use Storable Class Editor to create the fields. For instructions about making custom fields available to Java applications such as UE Mapper, see *BRM Developer's Guide*.

Displaying the Description of Custom Events

If you load events using PCM_OP_ACT_USAGE, include the field that describes your *custom* events in the event mapping by giving a default value for the field. By doing this, the description of your custom events is displayed in the Event Browser.

To display the description of custom events in the Event Browser, follow these steps:

1. Using UE Mapper, open the event import template used to import log files with custom usage events.
2. When creating the flist for event mapping, map the PIN_FLD_SYS_DESCR field of the custom event to a **default value**, where the value represents the description of the custom event.

If you load events using PCM_OP_LOAD_SESSION, the opcode fills in the PIN_FLD_SYS_DESCR field with a default value automatically. PCM_OP_LOAD_SESSION uses the default value **Session:event type**, where *event type* is the event type you specify in the **Event Properties** tab. If you do not want this description, map the field to a default value of your choice.

Adding Options to UE Mapper Drop-Down Lists

You might need to add items, such as delimiters, to some drop-down lists in the **Data Definition** tab of UE Mapper. The drop-down lists include popular options, but you can edit the lists in the **UEMapper.properties** file to include other options you need. Follow the instructions in the **UEMapper.properties** file in `/opt/portal/release/DevCenter/UEMapper.properties`.

Migrating Event Import Templates from One BRM Database to Another

Use the **pin_uei_deploy** utility to migrate an event import template from one database to another. For example, use the utility when you move a template from a test database to a production database. Using the **pin_uei_deploy** utility, you read an event import template from one database, save it as an output file on a local system, and then load it into another database.

Important: Do *not* modify the event import template when it is saved as a file on the local system. To modify an event import template, use UE Mapper in Developer Center.

To migrate an event import template from one database to another:

1. Ensure that BRM is running on both servers.
2. Ensure that you have a **pin.conf** file in the same directory as **pin_uei_deploy**.
3. Ensure that you have write privileges for the directory where you intend to save the output file.
4. Check the output directory to ensure that you do not overwrite output files you want to save. The **pin_uei_deploy** utility overwrites an output file of the same name without displaying a warning.
5. In the **pin_uei_deploy** utility **pin.conf** file, set the connection information to point to the database you are migrating the event import template *from*.
6. List all the event import templates stored on that database.

```
pin_uei_deploy -l
```

Note the name of the template you are migrating.

7. Download the template from the database to a local file. You can use any name for the output file. The output file is saved to the current directory unless you specify a path.

```
pin_uei_deploy -t template_name -r -o output_file_name
```

8. In the `pin_uei_deploy` utility `pin.conf` file, set the connection information to point to the database you are migrating the event import template *to*.

9. List all the event import templates stored in that database.

```
pin_uei_deploy -l
```

If the name of the template you are migrating exists in that database, ensure that you want to delete or overwrite the existing template as described in the next step.

10. Load the template using these options:

If the name of the template you are migrating does *not* already exist on the database, load the template using the *create* (`-c`) mode:

```
pin_uei_deploy -t template_name -c -i output_file_name
```

If the name of the template you are migrating already exists in the database, do one of the following:

- Delete the existing template in the database before loading using the `-c` mode:

```
pin_uei_deploy -t template_name -d
```

- Load the template using the *modify* (`-m`) mode of operation to overwrite the existing template:

```
pin_uei_deploy -t template_name -m -i output_file_name
```

11. Verify that the template has been loaded by listing all templates in the database:

```
pin_uei_deploy -l
```

Loading Events from External Sources

This chapter describes how to load event data from event log files into the Oracle Communications Billing and Revenue Management (BRM) database to be rated. It also provides troubleshooting information for loading events.

For general information about importing events, see "[About Rating Events Created by External Sources](#)".

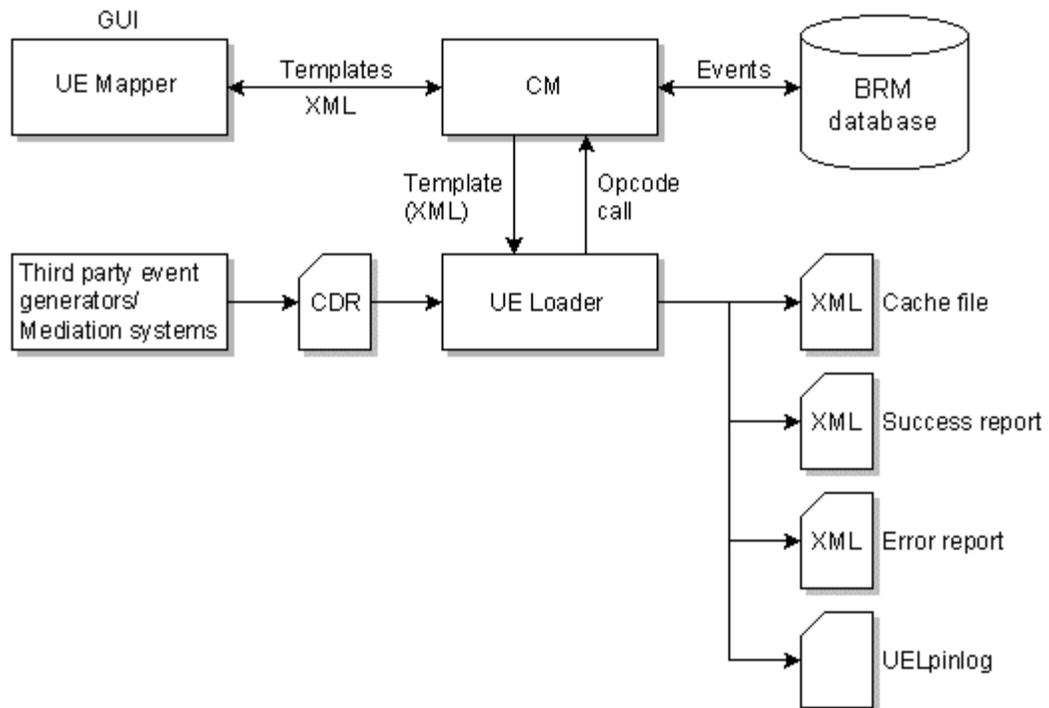
About Loading Events

You load data from an event log file into the BRM database by creating an event import template with UE Mapper. You then load the event data manually by using UE Loader or automatically by using Batch Controller and a batch handler.

UE Loader is a command-line utility that reads data about service usage from the event log file, reads the event import template for instructions on loading the data, such as which opcodes to call to load the data, and loads the data into the BRM database. When the events are loaded, BRM rates the events and changes customer account balances. You can set Batch Controller and create a UE Loader batch handler to run UE Loader automatically (to load event data on a regular basis). For information about Batch Controller, see "Controlling batch operations" in *BRM System Administrator's Guide*.

For information on creating an event import template and a technical overview of the event import process, see "[About Rating Events Created by External Sources](#)".

[Figure 22-1](#) shows how UE Mapper and UE Loader work to load events into the BRM database:

Figure 22-1 Loading Events into BRM Using UE Mapper and UE Loader

Configuring UE Loader

To configure UE Loader, edit the UE Loader properties file (*BRM_Home/apps/uel/Infranet.properties*). Follow these procedures:

- [Configuring Connection Information for UE Loader](#)
- [Specifying the Event Log File Location](#)
- [Specifying Whether to Log Duplicate Event Log Files](#)
- [Setting the Number of Threads for Performance](#)
- [Setting Memory Options](#)
- [Specifying How to Load Events](#)
- [Configuring Log File Locations for UE Loader](#)
- [Specifying the Maximum Load Errors to Process before Quitting](#)
- [Specifying the Directory to Store Logged Event Records](#)

Configuring Connection Information for UE Loader

To specify connection information:

1. Open UE Loader properties file (*BRM_Home/apps/uel/Infranet.properties*).
2. Follow the instructions in the file to set the standard connection information, such as the Connection Manager (CM) name and port number.
3. Save and close the file.

Enabling UE Loader to Convert Flists to Strings

Important: This is a mandatory configuration.

Ensure that UE Loader is configured to convert flists to strings:

1. Open UE Loader properties file (*BRM_Home/apps/uel/Infranet.properties*).
2. Ensure that the **infranet.flist.tostring.enable** entry is set to **true**:

```
infranet.flist.tostring.enable = true
```

If this entry is not present or is set to **false**, UE Loader might load records into the wrong accounts.

Specifying the Event Log File Location

When you run UE Loader, you specify only the name of the event log file, not the file location. You specify the file location in the **Infranet.properties** file.

For information about UE Loader log files, see "[Troubleshooting Event Loading](#)".

1. Open the properties file (*BRM_Home/apps/uel/Infranet.properties*).
2. Enter the directory name in the **infranet.uel.event_log_file_location** entry. This is the directory that stores event log files for loading.

When specifying the path, always use forward slashes (/), and type a slash after the last directory name.

```
infranet.uel.event_log_file_location=/var/portal/7.3.1/apps/uel/
```

3. Save and close the file.

Specifying Whether to Log Duplicate Event Log Files

By default, UE Loader does not notify you when a duplicate event log file is loaded into the database.

You can configure UE Loader to log an error when a duplicate event log file is loaded by using the **infranet.uel.duplicate_check** entry. When this entry is set to **True**, UE Loader creates a **DuplicateFiles.txt** file and adds the following for each duplicate file:

- The duplicate event log file's name
- The timestamp of when the duplicate event log file was processed

To log an error when duplicate event log files are loaded into the database:

1. Open UE Loader properties file (*BRM_Home/apps/uel/Infranet.properties*).
2. Change the **infranet.uel.duplicate_check** entry to **True**:

```
parameter infranet.uel.duplicate_check=True
```

3. Save and close the file.

Setting the Number of Threads for Performance

You can improve UE Loader performance by changing the maximum number of threads. Running multiple threads loads multiple events simultaneously, which gives faster performance but puts a greater load on the BRM system. The number of threads

that you specify depends on your BRM system configuration, for example, the number and speed of CPUs.

Note: The number of threads used when processing also depends on the event import template that you use to load the events. For example, if you load events based on their occurrence in the event log file, they are processed one at a time, so using multiple threads does not improve performance. For more information, see "[Specifying How to Load Events](#)".

1. Open the UE Loader properties file (*BRM_Home/apps/uel/Infranet.properties*).
2. Change the **infranet.uel.max_threads** entry, for example:

```
infranet.uel.max_threads=5
```

3. Save and close the file.

Tip: You can run multiple UE Loaders to improve performance. Each UE Loader can run multiple threads. If you have only one CM and one DM, however, there is a limit to how much you can improve performance by using multiple UE Loaders.

Setting Memory Options

Avoid memory errors with these guidelines:

- UE Loader can process large log files, such as files with 200,000 records. The memory required to load events depends on the number of records in your event log file.
- If you have a large number of records and UE Loader reports an out-of-memory error (error code **-2**), change the **infranet.uel.queue_size** entry in the **Infranet.properties** file. Setting this entry causes UE Loader to read records into memory in sizable chunks rather than reading all records into memory at once.

Important: If the **infranet.uel.queue_size** entry in the **Infranet.properties** file is set to **0**, all records are read into memory at once.

- You can configure the memory available to UE Loader by modifying the **-mx** setting in the **uel** script.

Specifying How to Load Events

You can load events based on their occurrence in the event log file, or you can load them based on their account identifier. Specify the loading method when you create the event import template by using the **Group events by account identifier** option in UE Mapper. For more information, start Developer Center and see the UE Mapper Help.

Important: If a custom bill item is rated cumulatively, UE Loader must group events by account identification for loading. For more information, see UE Mapper Help.

Configuring Log File Locations for UE Loader

To specify locations of log files:

1. Open the UE Loader properties file (*BRM_Home/apps/uel/Infranet.properties*).
2. Enter directory names in the following entries:
 - To set the location of the error file that records errors found in event loading mode, use the **infranet.uel.load_error_file_location** entry.
 - To set the location for the record of successfully loaded events, use the **infranet.uel.load_success_file_location** entry.
 - To set the location for the standard BRM client error log file, use the **infranet.uel.error_log_file_location** entry.
 - To set the location for the file that records events you filter by event field values, use the **infranet.uel.filter_log_file_location** entry.

When specifying the path, always use forward slashes (/), and type a slash after the last directory name.

```
infranet.uel.error_log_file_location=/var/portal/7.3.1/apps/uel/
```

3. Save and close the file.

Specifying the Maximum Load Errors to Process before Quitting

The maximum number of errors applies to errors generated from all threads, not errors per thread.

1. Open the properties file (*BRM_Home/apps/uel/Infranet.properties*).
2. Set the **infranet.uel.max_errors** entry, for example:

```
infranet.uel.max_errors=100
```

3. Save and close the file.

Specifying the Directory to Store Logged Event Records

You can set up filters so that particular records are logged to a file, whether you load them or not. See ["About Discarding Event Records Based on Record Content"](#) and ["About Logging Event Records Based on Record Content"](#).

To specify the directory to store the files that log filtered records:

1. Open the UE Loader properties file (*BRM_Home/apps/uel/Infranet.properties*).
2. Set the **infranet.uel.filter_log_file_location** entry.
3. Save and close the file.

Specifying the Default Date/Time Format for Timestamps

You can specify a timestamp format to support loading events from different countries that format dates and times in different ways. The timestamp format must match the

date/time format in the event log file that you are importing. The timestamp format is used by fields with the data type **PIN_FLDT_TSTAMP**.

You can specify the date/time format for each timestamp field (fields with the data type **PIN_FLDT_TSTAMP**) in your event log file when you create the event import template. This is necessary if you have records that use different timestamp formats in the same event log file.

If you do not specify the date/time format for a record on a per field basis when you create the event import template, UE Loader uses the date/time format specified in its **Infranet.properties** file as the default format.

To set the default date/time format:

1. Determine the date format by looking at the event log file you are importing.
2. Open the properties file (*BRM_Home/apps/uel/Infranet.properties*).
3. Change the **infranet.uel.date_pattern** entry to match the date format in the event log file you are importing. For example, if your records use a time format specified in a.m./p.m. hours (01 - 12), you specify hours by using a lowercase **h** and including the **a** parameter to indicate that the time is in a.m./p.m. format:

```
infranet.uel.date_pattern=dd/MMM/yyyy:hh:mm:ss a zzzz
```

See "Specifying Time Format Syntax".

4. Save and close the file.

Specifying Time Format Symbols

Table 22–1 shows examples of date formats:

Table 22–1 Example Date/Time Formats

Date in Event Log File	Date/time Format in Infranet.properties File
26/October/2005:10:55:49 Pacific Standard Time	dd/MMM/yyyy:hh:mm:ss a zzzz
12:08 PM	h:mm a
12 o'clock PM, Pacific Daylight Time	hh 'o'clock' a, zzzz
2005.07.10 AD at 15:08:56 PDT	yyyy.MM.dd G 'at' hh:mm:ss z
Wed, July 10, '05	EEE, MMM d, 'yy
0:00 PM, PST	K:mm a, z
2005.July.10 AD 12:08 PM	yyyyy.MMMMMM.dd GGG hh:mm aaa

Specifying Time Format Syntax

Each element in the time format syntax is represented by a character. For example, the year is represented by the "y" character. You control how each element is matched by the number of characters you use. For example, with the year 2001:

- yy matches 01
- yyyy matches 2001

There are also different symbol types, such as Text and Number. The number of characters has a different meaning for different symbol types as shown in Table 22–2:

Table 22–2 Time Format Syntax

Symbol Type	Rules for Number of Characters
Text	4 characters or more uses the full word. Fewer than 4 characters presents an abbreviation. For example, for E (day of the week): <ul style="list-style-type: none"> ▪ E = Wed ▪ EEEE = Wednesday
Number	Any number of characters uses the minimum required number of digits, except for years. With years, 2 characters uses the short version, for example: yy = 99
Text and number	3 characters or more uses text, 1 or 2 characters uses numbers. For example: <ul style="list-style-type: none"> ▪ MMM = July ▪ MM = 07

In addition, you can use an escape character to match text as shown in [Table 22–3](#):

Table 22–3 Use of Escape Character to Match Text

Log File	Date/Time Symbols
July 2 at 12:01	MMM d 'at' hh:mm

To match a single quotation mark, use double-single quotation marks as a literal, as shown in [Table 22–4](#):

Table 22–4 Use of Quotation Marks to Match Text

Log File	Date/Time Symbols
July 2 "99	MMM d "yy

Any characters outside the range of a-z and A-Z, such as colon (:), semicolon (;), and at symbol (@), are treated as quoted text, even if they are not enclosed in single quotation marks.

Important: Characters within the ranges of a-z and A-Z that are *not* used as symbols, such as C, and are not enclosed in quotes, cause an error.

Allowing for Leniency in Time References

When a CDR contains a date value that uses a +1300 time reference, the system cannot read the date properly and returns an error. For example, the date entry might look like this: 20061202000000+1300.

To enable the system to process CDRs that use the +1300 time zone reference, set the **Infranet.uel.lenient** value to **true** in the UE Loader properties file (*BRM_Home/apps/uel/Infranet.properties*) The default value is **false**:

```
Infranet.uel.lenient = true
```

Configuring UE Loader for International Log Files

This section provides instructions for configuring UE Loader to use international log files.

Configuring the UE Loader `Infranet.properties` File

UE Loader supports English, French, German, and Japanese log files. To support these languages, set the following parameters in the UE Loader `Infranet.properties` file (`BRM_Home/apps/uel/Infranet.properties`) shown in [Table 22-5](#):

Table 22-5 Parameters in UE Loader File

Infranet.properties File Entry	Description
<code>infranet.uel.event_log_file_lang_code</code>	Specify the language code. See "Language, Country, and Encoding Codes".
<code>infranet.uel.event_log_file_country_code</code>	Specify the country code. UE Loader supports all country codes that apply to each language (English, French, German, and Japanese).
<code>infranet.uel.event_log_file_variant_code</code>	Specify the variant code.
<code>infranet.uel.event_log_file_encoding</code>	Specify the encoding code. See "Language, Country, and Encoding Codes".
<code>infranet.uel.date_pattern</code>	Specify the date pattern for the locale. See "Specifying the Default Date/Time Format for Timestamps".

Language, Country, and Encoding Codes

Listed below are the language, country, and encoding codes for languages supported by each operating system.

Note: UE Loader supports all country codes for each language.

[Table 22-6](#) lists the system encoding support for English.

Table 22-6 System Encoding Support for English

Operating System	Language Code	Country Codes	Encoding Code
Solaris	en	AU, CA, IE, NZ, ZA, GB, and US	8859-1
Linux	en	AU, CA, IE, NZ, ZA, GB, and US	None
HP-UX IA64	en	AU, CA, IE, NZ, ZA, GB, and US	ISO81
AIX	en	AU, CA, IE, NZ, ZA, GB, and US	ISO8859-1

[Table 22-7](#) lists the system encoding for Japanese.

Table 22-7 System Encoding Support for Japanese

Operating System	Language Code	Country Codes	Encoding Code
Solaris	ja	JP	SJIS
Linux	ja	JP	None
HP-UX IA64	ja	JP	sjis
AIX	ja	JP	IBM-943

Table 22-8 lists the system encoding support for German.

Table 22-8 System Encoding Support for German

Operating System	Language Code	Country Codes	Encoding Code
Solaris	de	AT, DE, LU, and CH	8859-1
Linux	de	AT, DE, LU, and CH	None
HP-UX IA64	de	AT, DE, LU, and CH	ISO81
AIX	de	AT, DE, LU, and CH	ISO8859-1

Table 22-9 lists the system encoding for French.

Table 22-9 System Encoding Support for French

Operating System	Language Code	Country Codes	Encoding Code
Solaris	fr	BE, CA, FR, LU, and CH	8859-1
Linux	fr	BE, CA, FR, LU, and CH	None
HP-UX IA64	fr	BE, CA, FR, LU, and CH	ISO81
AIX	fr	BE, CA, FR, LU, and CH	ISO8859-1

Loading Events

You can load events manually by running UE Loader from a command line or you can schedule UE Loader to run automatically. See:

- [Loading Events Manually](#)
- [Loading Events Automatically](#)

About the Order of Events

If any event balance impacts are configured to start immediately and never expire, consider the order of events when you load them.

When an event is loaded, BRM checks the validity periods of the account's resource sub-balances. The sub-balance that has a start time earlier than the event's start time is the one that receives the impact. If there is no sub-balance that has an earlier start time, BRM creates a new sub-balance, sets its start time to the event's start time, and impacts that sub-balance. This can cause the creation of multiple sub-balances for the same resource when events are loaded out of order. Having multiple sub-balances for the same resource can impact performance.

For more information, see "[How Resources in Validity-Based Sub-Balances Are Updated](#)".

To prevent the creation of multiple account sub-balances for resources that start immediately and never expire, be sure to sort the events in ascending order based on their timestamp before running UE Loader.

Loading Events Manually

Use UE Loader to manually load events from data log files into the BRM database. UE Loader is a command-line utility installed with BRM, and its executable file is in *BRM_Home/apps/uel*. For information on UE Loader command-line syntax, see "[Universal Event Loader](#)".

For information on loading events from data log files automatically, see "[Loading Events Automatically](#)".

Before You Begin

- You must configure UE Loader before you can load events. See "[Configuring UE Loader](#)".
- To avoid memory errors, follow the guidelines in "[Setting Memory Options](#)".
- An event import template is required to load event data. You must know the exact name of the event import template that was created for the event log file you are loading.
- Before loading events, you must specify the date/time format used in the event log file. You can specify the date/time format for all records in your log file by editing the UE Loader **Infranet.properties** file. See "[Specifying the Default Date/Time Format for Timestamps](#)". The date/time format for each record can also be specified on a per field basis when the event import template is created. If the date/time format is not specified on a per field basis, the date/time format specified in the **Infranet.properties** file is used.
- Before loading events, run the UE Loader program on a test database to test the event import template and any custom opcodes that were created for loading your log file events.

Instructions

To load events manually, run UE Loader by using this syntax:

```
uel -t template_name [-v] [-m parse|load] [-test] log_file_name
```

The optional **-v** verbose parameter causes UE Loader to output error messages or debug messages on the console according to the debug level you set in the **Infranet.properties** file.

The **-m** and **-test** modes are used more during testing of event loading. When you are loading events into a production database, you do not need to use these options.

For example, if the template name is *WebServer* and the event log file name is *webserver.log*, use this command to load events:

```
uel -t WebServer webserver.log
```

Notes

- Do not enter the path of the event log file. You specify the path in the UE Loader **Infranet.properties** file. See "[Specifying the Event Log File Location](#)".
- The log file can use any file extension or no extension.

For information on UE Loader command-line syntax, see "[Universal Event Loader](#)".

Loading Events Automatically

You can load event log files automatically and manually. Some BRM service integration components provide a batch handler that executes UE Loader to process event log files automatically, while BRM system software provides Batch Controller that controls the handler. You can also write a custom handler. You can load files of event records automatically on a periodic basis by configuring UE Loader, Batch Controller, and a handler.

To set up automatic event loading:

1. Configure UE Loader. See ["Configuring UE Loader"](#).
2. Set up a handler to launch UE Loader.
See the documentation of the service integration component you use for instructions on setting up the handler for that component.
3. Set up Batch Controller to do these tasks:
 - a. Scan a configured directory for record files from an external source.
 - b. Move the files to the appropriate handler for processing.

See also "Controlling Batch Operations" in *BRM System Administrator's Guide*.

Troubleshooting Event Loading

This section explains UE Loader log files and error codes, and provides tips on troubleshooting event loading failures.

Understanding UE Loader Log Files

UE Loader records errors and successes in the appropriate log files in *BRM_Home/apps/uel*. [Table 22-10](#) describes the UE Loader log files.

Table 22-10 UE Loader Log Files

Type of File	Log File Name	Description
Cache file	<i>event_log_file_name_cache.xml</i>	<p>The cache file is an intermediate file generated by UE Loader after it parses the event log file and before it loads events into the database. UE Loader stores the results of event log parsing in the cache file, which includes:</p> <ul style="list-style-type: none"> ■ The event records ■ The customer accounts associated with the events ■ A record number for each event record <p>The record number appears in the error log so you know which record to fix in the cache file if there are errors during event loading.</p>

Table 22–10 (Cont.) UE Loader Log Files

Type of File	Log File Name	Description
Filter log	<i>event_log_file_name_filtered.xml</i>	The filter log file is generated when events are loaded. It lists records you have specified to archive. Records are archived based on filters you set up in the event import template. See " About Logging Event Records Based on Record Content ". Note: The event import template specifies whether filtered records are loaded into the BRM database.
Error log	<i>event_log_file_name_lerr.xml</i>	The error log file is generated when events are loaded. It reports any loading errors and specifies the records in the event log file that caused the errors.
Success log	<i>event_log_file_name_lsucc.xml</i>	The success log file is generated when events are loaded. It lists all records that load into the database successfully.
Application log	uel.pinlog	The standard application log file contains complete descriptions of the errors generated in the other log files. To log all errors, set the log level to 3.

When errors occur, you typically check the log files in this order:

1. UE Loader pinlog
2. Error file
3. Success file
4. Cache file

Check the cache file for NULL account attributes if an account could not be found during event loading.

5. Filter file

Check the filter file if you have set up filters for event loading to see that events were loaded and logged according to the filter.

Note: The name that UE Loader generates for its log files includes the extension of the event log file. For example, when the event log file is named **CDR3.txt**, the cache, filter, error, and success files are called:

- **CDR3.txt_cache.xml**
 - **CDR3.txt_filtered.xml**
 - **CDR3.txt_lerr.xml**
 - **CDR3.txt_lsucc.xml**
-

Understanding UE Loader Error Codes

Table 22–11 describes the UE Loader error codes that appear in the **uel.pinlog**, *event_log_file_name_lerr.xml* file, and *event_log_file_name_cache.xml* file:

Table 22–11 UE Loader Error Codes

Error Code	Brief Description	Error Description	Possible Required Actions
0	NO_ERROR	All records loaded successfully.	No action needed.
-1	FAILED_HEADER_TRAILER_CHECK	The header record or trailer record check failed.	<ul style="list-style-type: none"> ■ If the content does not match the header and trailer check, check for a file content problem. ■ Use UE Mapper to ensure that the header or trailer check is set up correctly in the event import template.
-2	NOT_ENOUGH_MEMORY	UE Loader ran out of memory.	<ul style="list-style-type: none"> ■ Reconfigure the memory available to UE Loader by modifying the -mx setting in uel script. ■ Reconfigure the queue size in the Infranet.properties file. For details, see "Setting Memory Options".
-3	OUT_OF_DISK_SPACE	<p>UE Loader ran out of disk space while creating log files.</p> <p>For example, there was insufficient disk space in the specified directory to store UE Loader success, error, or filter log files.</p>	Specify a location for log files that contains more disk space. See " Configuring Log File Locations for UE Loader ".
-4	TEMPLATE_ERROR	<p>UE Loader detected a problem with the event import template.</p> <p>For example, you used the wrong version of a template (a template that was created with an earlier version of UE Mapper).</p>	To convert an event import template to the latest version, migrate the template in UE Mapper. See the UE Mapper Help for instructions.
-100	BAD_INF_CONNECTION	<p>The BRM system is not available.</p> <p>For example, the CM is down.</p>	<ul style="list-style-type: none"> ■ Start BRM if it is down. ■ Check your connection settings, such as the port number. See "Configuring Connection Information for UE Loader".

Table 22–11 (Cont.) UE Loader Error Codes

Error Code	Brief Description	Error Description	Possible Required Actions
1	FIELD_PARSE_ERROR	<p>An error occurred when parsing the fields in the event log file.</p> <p>For example, UE Loader detected no matching string delimiter when parsing the event record data.</p>	<ol style="list-style-type: none"> 1. Open the cache file. 2. Find the record with the error code 1. 3. Correct the error in the record. 4. Save and close the cache file. 5. Reload the events by using the -m load option. See "Reloading Events That Failed to Load".
2	ACCT_FORMAT_ERROR	<p>When UE Loader constructed the input flist to pass to the opcode that finds the account, it detected an error in the data format.</p> <p>Possible causes:</p> <ul style="list-style-type: none"> ■ Incorrect field specification. For example, the login is specified as fields 1 through 5 when the login should be specified as fields 1 through 7. ■ Incorrect setup of the input flist mapping for finding accounts associated with events. 	<ul style="list-style-type: none"> ■ If the field specification is incorrect, open the event log file and correct the record. ■ If the input flist mapping for finding accounts is incorrect, use UE Mapper to correct the mapping in the event import template.
3	LOAD_FORMAT_ERROR	<p>When UE Loader constructed the input flist to pass to the opcode that loads the event data into BRM, it detected an error in the data format.</p> <p>Possible causes are similar to error code 2.</p>	<ol style="list-style-type: none"> 1. Open the cache file. 2. Find the record with error code 3. 3. Correct the error in the record. 4. Save and close the cache file. 5. Reload the events by using the -m load option. See "Reloading Events That Failed to Load".

Table 22–11 (Cont.) UE Loader Error Codes

Error Code	Brief Description	Error Description	Possible Required Actions
4	PREV_PARSE_ERROR	<p>When loading events, a parse error occurred.</p> <p>Note: You do not receive this error if you choose to continue loading events when parse errors occur. To configure this, set the infranet.uel.load_on_parse_errors entry in the UE Loader Infranet.properties file. You can also configure a maximum number of parse errors before UE Loader aborts.</p>	<p>Correct the record in the event log file.</p> <p>Note: This type of error cannot be corrected in the cache file (for reloading with the -m load option) because the order of events is incorrect. The error must be corrected in the event log file.</p>
5	TOO_MANY_LOAD_ERRORS	<p>The number of load errors exceeded the number of errors you allow. This error is generated when a number of load format errors (code 3) or load opcode errors (code 101) occur.</p> <p>Note: To set the maximum number of errors, use the infranet.uel.max_load_errors entry in the UE Loader Infranet.properties file.</p>	<p>See the possible actions for error codes 3 and 101.</p>
10	COMMAND_LINE_ARG_ERROR	<p>The command line syntax you used to run UE Loader is incorrect.</p>	<p>See the "Universal Event Loader" utility page for the correct syntax.</p>
11	FILE_NOT_FOUND_ERROR	<p>The event log file you specified in the UE Loader command line cannot be found.</p>	<ul style="list-style-type: none"> ■ Ensure that the location of the event log file is specified correctly in the Infranet.properties file. ■ Use the correct name for the event log file in the UE Loader command.
100	ACCT_NOT_FOUND	<p>The opcode that finds accounts associated with event data could not find an account.</p> <p>Possible causes:</p> <ul style="list-style-type: none"> ■ The account does not exist in BRM. ■ The account exists in BRM, but the login or service specified is incorrect. ■ The input flist mapping for finding accounts associated with event data is incorrect. 	<ul style="list-style-type: none"> ■ Correct the record in the event log file. ■ Use UE Mapper to ensure that the correct service type associated with the event is specified in the event import template. ■ If the input flist mapping for finding accounts is incorrect, use UE Mapper to correct the mapping in the event import template.

Table 22–11 (Cont.) UE Loader Error Codes

Error Code	Brief Description	Error Description	Possible Required Actions
101	LOAD_OPCODE_ERROR	The opcode that loads event data into BRM could not load the data.	<ul style="list-style-type: none"> ■ Open the cache file and correct the record. ■ If the input flist mapping for loading event data is incorrect, use UE Mapper to correct the mapping in the event import template.
111	The template61.dtd file is missing.	The default UE Loader template file (template61.dtd) is not present in the directory from which the UE Loader utility is run.	Copy the template file to the directory from which you run UE Loader. The default template, template61.dtd , is in <i>BRM_Home/apps/uel</i> .

Troubleshooting Tips

Some common causes of failure to load event records into the BRM database:

- Incorrect connection settings in your UE Loader configuration file.
See the UE Loader properties file (*BRM_Home/apps/uel/Infranet.properties*) for instructions on setting each entry.
- Incorrect format of the event data in your event log files.
UE Loader expects the records in your event log file to use the format defined in the event import template.
- Incorrect setup of the event import template used to import the event data.
UE Loader reads the event import template to obtain instructions for loading the events, so the event import template must specify the correct mapping for the records in your log file.
- Incorrect time format of the event data.

Specifying the Maximum Load Errors to Process Before Quitting

If you choose to continue loading events when parse errors occur, you can specify the number of errors to allow before UE Loader quits. To set this maximum number of load errors, use the **infranet.uel.max_load_errors** entry in the UE Loader properties file (*BRM_Home/apps/uel/Infranet.properties*).

Troubleshooting Connectivity Errors

If you have trouble connecting to the database, ensure that you are using the correct host name, and that the port number, database number, and host name are specified correctly in your UE Loader properties file (*BRM_Home/apps/uel/Infranet.properties*).

Troubleshooting Preprocessing Errors

UE Loader reports two types of preprocessing errors:

- Errors found when reading the event log file.
These errors include the record number, line number, and error description. These errors are typically the result of UE Loader finding an unexpected type of data, such as unmatched literal string indicators.

- Errors in the custom opcode that identifies the account that generated the event.

Some preprocessing errors can be fixed by editing the event import template. Errors in the template are typically fixed by a developer by using the UE Mapper application in Developer Center.

In some cases, an event log file includes a record that has not been generated correctly by the event log file source. In this case, edit the event log file manually to fix the record before loading it.

Troubleshooting Event Loading Errors

Event loading errors include the record number and contents of the record as read from the intermediate cache file. To troubleshoot loading errors, open the **uel.pinlog** file and read the contents of the error buffer returned by the BRM opcode.

Loading errors are usually the result of the following:

- Errors in the BRM system, such as an offline Data Manager (DM).
- Errors in the event import template. Errors in the template are typically fixed by a developer familiar with BRM programming concepts and techniques.

Testing Event Importing

During testing of event importing, note that:

- UE Loader does not use the time set by the **pin_virtual_time** utility. If the event start and end times are not obtained from the log file, UE Loader uses the system time.
- You may want to load a specific event log file more than once.

If you load events from an event log file successfully, UE Loader generates the log files for it, such as the success log file and the error log file. These log files must be deleted before you can reload events from the same event log file.

Reloading Events That Failed to Load

If events fail to load into the BRM database, UE Loader logs the records in an error log file called `event_log_file_name_lerr.xml`. Errors are also recorded in the `event_log_file_name_cache.xml`.

1. Find the record number of the event record(s) that failed to load by looking in the `event_log_file_name_lerr.xml` file.
2. Find the event records that failed to load in the `event_log_file_name_cache.xml` file by using the record number(s) you found in the error log file.
3. Fix the errors in the event records.
4. Reload the events by running UE Loader with the **-m load** parameter:

```
uel -t template_name -m load event_log_file_name
```

UE Loader reads the files generated in the previous run, and ignores previously loaded events.

For more information on the cache file and error file, see "[Understanding UE Loader Log Files](#)".

Improving UE Loader Performance

You can improve UE Loader performance by adjusting the number of UE Loader threads and processes and by modifying entries in the UE Loader **Infranet.properties** file.

Important: Optimal performance configuration for UE Loader depends on your system configuration such as the number and speed of CPUs and the size of your hard drive and cache memory. To find the UE Loader settings that work best for your system, you must test different combinations. For example, vary the number of threads and processes and test with different UE Mapper templates, record distribution, and queue sizes.

You can improve UE Loader performance by doing one or more of the following:

- Increase the number of threads running in UE Loader.

Increase the number of threads in the **infranet.uei.max_threads** entry in the **Infranet.properties** file. For more information, see ["Setting the Number of Threads for Performance"](#).

Important: If the loading order of records is important, for example, when loading session events, use multiple threads *only* if you configure UE Mapper to load CDRs grouped by account ID.

- Increase the number of UE Loader processes.

You can run several UE Loader processes at the same time and load a file with each process. Do this by splitting the input file into several files before loading.

Even better performance can be achieved by running several UE Loader processes using multiple threads in each process.

- Limit the number of records loaded into memory.

Increase the queue size in the **infranet.uel.queue_size** entry in the **Infranet.properties** file. The **queue_size** entry specifies the number of records to load simultaneously. The queue size is shared by multiple threads, which have concurrent access to the queue. For more information, see ["Setting Memory Options"](#).

- Configure UE Loader to load CDRs grouped by account ID.

Select **Group events by account identifier** when setting up your template in UE Mapper. For more information, start Developer Center and see UE Mapper Help.

- Load CDRs in the order they occur when using multiple threads.

Select **Group events by account identifier** in UE Mapper and ensure that all CDRs from the same account are loaded in order (ordered by time).

Retrieving Data about Events You Load

BRM stores information about batches of events that you load in a **batch/gel** object. This object contains information such as the event log file name and the date and time loaded.

Note: You can display the event object in Event Browser.

Part IV

Setting Up Rerating

Part IV describes how to configure rerating, which is the process used to rerate events originally rated in batch by Pipeline Manager or in real time. This part covers the different tools used in rerating and the situations in which you use them.

Part IV contains the following chapters:

- [About Rerating Events](#)
- [About Real-Time Rerating Pipelines](#)
- [About Rerating Pipeline-Rated Events](#)
- [Using Event Extraction Manager](#)
- [Configuring Rerating in Pipeline Manager](#)
- [About Comprehensive Rerating Using pin_rerate](#)
- [Configuring Comprehensive Rerating](#)
- [Using the pin_rerate Utility](#)

About Rerating Events

This chapter provides an overview of Oracle Communications Billing and Revenue Management (BRM) rerating.

About Rerating

BRM rerating is the process you use to rerate events that were originally rated in batch by Pipeline Manager or in real time. You might rerate events for several reasons:

- To rerate a group of accounts after changing a price plan
- To rerate an account when a customer service representative (CSR) backdates a subscriber's purchase or cancellation
- To back out events when call detail records (CDRs) contain field errors
- To rerate events when the rating conditions change during the session

Note:

- Member, child, and subordinate accounts in charge sharing, discount sharing, and account sponsorship and hierarchy groups are not automatically rerated when the parent or sponsoring account is rerated. For more information, see "[About Rerating Events for Account Sharing Groups](#)".
 - You can rerate remittance accounts, but rerating subscriber accounts does not automatically rerate the remittance accounts. Events already included in remittance processing are not rerated.
-
-

The balance impact of rerating is the difference between the original rated event and the rerated event. This difference is applied to the account balance in the current billing cycle. For more information, see "[How BRM Applies the Balance Impacts of Rerating](#)".

About the Rerating Process

The rerating process consists of the following steps:

1. **Extracting events.** BRM extracts the events to rerate for an account or a group of accounts from the BRM database.
2. **Restoring balances.** BRM backs out the original event balance impacts and restores the account's discount and aggregation balances to their original states before rating the events.

3. **Calculating new balances.** BRM rerates the events based on new pricing information and calculates the new discount and aggregation balances.
4. **Recording the event.** BRM loads the events with the new balance impacts into the BRM database, updating all relevant account data.

For more information on configuring rerating with your charging engine:

- If you use Pipeline Manager, see ["Configuring Rerating in Pipeline Manager"](#).
- If you use Oracle Communications Elastic Charging Engine (ECE), see ["Configuring BRM for Elastic Charging Engine Rerating"](#).

About Automatic Allocation from Rerating

By default, BRM creates unallocated items for any adjustment items that are created as a result of rerating.

Corrective Billing and Automatic Allocation of Rerating Adjustments

If your corrective bills must contain the aggregation and allocation of automatic adjustments to the items on the original bill, you must enable the **AllocateReratingAdjustments** business parameter before you rerate the original bills. When the **AllocateReratingAdjustments** business parameter is enabled, adjustment details by original item by original bill are reported on the next bill for regular billing also.

To ensure that the corrective billing process includes or excludes these adjustments, check the setting for the **AllocateReratingAdjustments** business parameter *before* you run the rerating process.

If BRM has rerated a **/bill** object without allocating automatic adjustments (from the rerating) to the original bills, the corrective bill for that **/bill** object will not include the item adjustments generated by that rerating.

Note: For such a **/bill** object, if you set the **AllocateReratingAdjustments** business parameter to **enabled** and rerun the rerating process for the bill, BRM does not allocate the adjustments.

You must manually allocate the rerated items *before* you generate the corrective bill for such a **/bill** object.

If you enable BRM to automatically allocate the item adjustments generated by the rerating process to the original bill, BRM allocates adjustments to the bill items being corrected in the following manner:

- For open item accounting, the adjustment items are allocated to each bill that was corrected. Allocation is made to each of the original bills that included events or items that were corrected by these adjustments.
- For balance forward accounting, corrections are posted to the last bill only. Therefore, the rerating allocation is made to the final bill only. This final bill carries over the balances from all prior bill periods.

Enabling Automatic Allocation of Rerating Adjustments

To enable automatic allocation of rerating adjustments:

1. Go to the *BRM_Home/sys/data/config* directory, where *BRM_Home* is the directory in which BRM is installed.
2. Run the following command, which creates an editable XML file from the **rerate** instance of the */config/business_params* object:

```
pin_bus_params -r BusParamsRerate bus_params_rerate.xml
```

This command creates the XML file named **bus_params_rerate.xml.out** in your working directory. To place this file in a different directory, specify the path as part of the file name.

3. Open the **bus_params_rerate.xml.out** file.
4. Search for the following line:


```
<AllocateReratingAdjustments>disabled</AllocateReratingAdjustments>
```
5. Change **disabled** to **enabled**.
6. Save the file as **bus_params_rerate.xml**.
7. Go to the *BRM_Home/sys/data/config* directory, which includes support files used by the **pin_bus_params** utility.
8. Run the following command, which loads this change into the appropriate */config/business_params* object:

```
pin_bus_params PathToWorkingDirectory/bus_params_rerate.xml
```

where *PathToWorkingDirectory* is the directory in which **bus_params_rerate.xml** resides.

Caution: BRM uses the XML in this file to overwrite the existing **rerate** 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 **rerate** configuration.

Note: To run this command from a different directory, see "pin_bus_params" in *BRM Developer's Guide*.

9. Read the object with the **testnap** utility or Object Browser to verify that all fields are correct.

See "Using testnap" in *BRM Developer's Guide* for general instructions on using the **testnap** utility. See "Reading Objects by Using Object Browser" in *BRM Developer's Guide* for information on how to use Object Browser.
10. Stop and restart the Connection Manager (CM). For more information, see "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.
11. (Multischema systems only) Run the **pin_multidb** script with the **-R CONFIG** parameter. For more information, see "pin_multidb" in *BRM System Administrator's Guide*.

About Deferred Taxes During Rerating

By default, BRM does not compute deferred tax during rerating.

Enabling Calculation of Deferred Taxes During Rerating

Note: When deferred taxation is enabled for rerating:

- You cannot use selective rerating.
 - You cannot rerate events created by Bill Now.
 - You cannot rerate events that use multiple tax suppliers.
-
-

By default, BRM calculates taxes on any deferred taxable amount in the rerated events during the subsequent bill run. The rerated tax appears on the invoice for the subsequent bill run. Calculating taxes during rerating using the **ApplyDeferredTaxDuringRerating** business parameter enables corrected invoices to show the corrected tax amount. When combined with corrective invoicing, rerated events will occur and their tax amounts will appear on the invoice for the original billing period.

To enable calculation of deferred taxes during rerating:

1. Go to the *BRM_Home/sys/data/config* directory.
2. Run the following command, which creates an editable XML file from the **rerate** instance of the */config/business_params* object:

```
pin_bus_params -r BusParamsRerate bus_params_rerate.xml
```

This command creates the XML file named **bus_params_rerate.xml.out** in your working directory. To place this file in a different directory, specify the path as part of the file name.

3. Open the **bus_params_rerate.xml.out** file.
4. Search for the following line:

```
<ApplyDeferredTaxDuringRerating>disabled</ApplyDeferredTaxDuringRerating>
```
5. Change **disabled** to **enabled**.
6. Save the file as **bus_params_rerate.xml**.
7. Go to the *BRM_Home/sys/data/config* directory, which includes support files used by the **pin_bus_params** utility.
8. Run the following command, which loads this change into the appropriate */config/business_params* object:

```
pin_bus_params PathToWorkingDirectory/bus_params_rerate.xml
```

where *PathToWorkingDirectory* is the directory in which **bus_params_rerate.xml** resides.

Caution: BRM uses the XML in this file to overwrite the existing **rerate** 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 rerate configuration.

Note: To run this command from a different directory, see "pin_bus_params" in *BRM Developer's Guide*.

9. Read the object with the **testnap** utility or Object Browser to verify that all fields are correct.

See "Using testnap" in *BRM Developer's Guide* for general instructions on using the **testnap** utility. See "Reading Objects by Using Object Browser" in *BRM Developer's Guide* for information on how to use Object Browser.

10. Stop and restart the CM. For more information, see "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.
11. (Multischema systems only) Run the **pin_multidb** script with the **-R CONFIG** parameter. For more information, see "pin_multidb" in *BRM System Administrator's Guide*.

About Rerating Events by Using the Rates Applied when the Rating Conditions Change During the Session

By default, events are rerated in order of occurrence based on the event end time. Rerating the events using the **OfferEligibilitySelectionMode** business parameter enables to rerate the events in order of event start time when the rating conditions change during the session. For example, if during a call session, the subscriber adds the called number of that session to a Friends and Family list, BRM applies the Friends and Family discount for the session from the time the called number is added to the Friends and Family list. When the call is rerated, the actual rate is applied from the start time of the call until the called number is added to the Friends and Family list, and the discount is applied for the remaining session.

Enabling Rerating when the Rating Conditions Change During the Session

To enable rerating when the rating conditions change during the session:

1. Go to the *BRM_Home/sys/data/config* directory, where *BRM_Home* is the directory in which BRM is installed.
2. Run the following command, which creates an editable XML file from the **rerate** instance of the */config/business_params* object:

```
pin_bus_params -r BusParamsRerate bus_params_rerate.xml
```

This command creates the XML file named **bus_params_rerate.xml.out** in your working directory. To place this file in a different directory, specify the path as part of the file name.

3. Open the **bus_params_rerate.xml.out** file.
4. Search for the following line:

```
<OfferEligibilitySelectionMode>endtime</OfferEligibilitySelectionMode>
```
5. Change **endtime** to **timeperiod**.
6. Save the file as **bus_params_rerate.xml**.
7. Go to the *BRM_Home/sys/data/config* directory, which includes support files used by the **pin_bus_params** utility.

8. Run the following command, which loads this change into the appropriate `/config/business_params` object:

```
pin_bus_params PathToWorkingDirectory/bus_params_rerate.xml
```

where *PathToWorkingDirectory* is the directory in which `bus_params_rerate.xml` resides.

Caution: BRM uses the XML in this file to overwrite the existing `rerate` 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 `rerate` configuration.

Note: To run this command from a different directory, see "pin_bus_params" in *BRM Developer's Guide*.

9. Read the object with the `testnap` utility or Object Browser to verify that all fields are correct.
See "Using testnap" in *BRM Developer's Guide* for general instructions on using the `testnap` utility. See "Reading Objects by Using Object Browser" in *BRM Developer's Guide* for information on how to use Object Browser.
10. Stop and restart the CM. For more information, see "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.
11. (Multischema systems only) Run the `pin_multidb` script with the `-R CONFIG` parameter. For more information, see "pin_multidb" in *BRM System Administrator's Guide*.

Understanding the BRM Rerating Features

Using BRM, you can rerate events in the following ways:

- **By using Pipeline Manager to rerate pipeline-rated events only.** This method is used for rerating events originally rated by Pipeline Manager.

With this method, you extract the events for rerating from the BRM database by using the `pin_event_extract` utility. You then rerate the events in a batch rerating pipeline.

Events are rerated and recorded using a two-step process: Events are rerated by the pipeline and then recorded in the BRM database when they are loaded by Rated Event (RE) Loader. As a result, pipeline event balance impacts are applied when they are recorded in the BRM database (event creation time).

Because an account can also have real-time event balance impacts that are recorded in real time (event end time), there is a possibility that the pipeline-rated and real-time rated event balance impacts are applied out of sequence, which might affect how BRM bills for the usage.

Note: When using Pipeline Manager to rerate pipeline-rated events, you must determine whether this method of rerating is sufficient to provide consistent results. Accounts with pipeline-rated usage may also have real-time events, such as purchase events, cancel events, or cycle fee events that can impact the rating of usage events.

- **By using the `pin_rerate` utility to rerate pipeline-rated and real-time rated events.** This method can be used for rerating events originally rated in real time and events originally rated in batch by Pipeline Manager.

With this method, you select events for rerating and rerate the events by running the `pin_rerate` utility. Pipeline-rated events are rerated in a real-time rerating pipeline and real-time rated events are rerated by real-time rating opcodes.

Pipeline-rated and real-time rated events are rerated and recorded in the same process. Events are rerated in order, based on the time they occurred (the event end time). This results in pipeline-rated *and* real-time-rated event balance impacts being applied in the correct sequence. For more information, see "[About Rerating Pipeline and Real-Time Events Concurrently](#)".

If you do not use Pipeline Manager for batch rating, you can also use `pin_rerate` to rerate only real-time-rated events.

Note: Using `pin_rerate` to rerate events can degrade system performance, depending on the system load.

When you have a high volume of events to rerate (for example, more than 100,000), you might be able to use Pipeline Manager to rerate pipeline-rated events and use `pin_rerate` to rerate real-time rated events to reduce the system load. You can do this if the accounts being rerated have only real-time-rated events or have only pipeline-rated events. If accounts have both real-time rated and pipeline-rated events, you should use `pin_rerate` to ensure consistent results.

For more information about rerating by using `pin_rerate`, see "[About Comprehensive Rerating Using `pin_rerate`](#)".

About Rerating Pipeline and Real-Time Events Concurrently

When you use `pin_rerate` to rerate pipeline-rated and real-time-rated events concurrently, by default, events are rerated in order of occurrence based on the event end time and recorded in the BRM database as they are processed. This process applies the pipeline and real-time event balance impacts in the sequence that the original real-time usage occurred.

For example, R_1 and R_2 are real-time-rated events and P_1 and P_2 are pipeline-rated events. The sequence in which these events were generated in real time was R_1, P_1, P_2, R_2 . However, the order in which they were recorded in the BRM database was R_1, R_2, P_1, P_2 . When these events are rerated by `pin_rerate`, the events are rerated and recorded in the original sequence in which they occurred (R_1, P_1, P_2, R_2).

You can also use `pin_rerate` parameters to rerate the events in the order they were originally recorded (R_1, R_2, P_1, P_2).

What You Can Do with Rerating

When using Pipeline Manager to rerate pipeline-rated events only, you can do the following:

- Select accounts and events for rerating based on various criteria, such as:
 - Account number
 - Accounts that have events related to specific deals, products, discounts, or specific event and service types
 - Events associated with specific products, discounts, subscription services, or bill units
- Perform back-out-only rerating, which backs out the balance impacts of rating without reapplying new balance impacts.

When using **pin_rerate** to rerate pipeline-rated and real-time-rated events, you can do the following:

- Select accounts and events for rerating based on various criteria such as account number, deals, products, discounts, event type, service type, bill unit and so on.
Rerating a bill unit rerates all usage events, cycle events, billing-time discounts, folds, and rollover events associated with the bill unit.
- Perform back-out-only rerating, which backs out the balance impacts of rating without reapplying new balance impacts.
- Rerate real-time-rated and pipeline-batch-rated events concurrently or, if you do not use Pipeline Manager for batch rating, rerate only real-time-rated events.
- Assign a rerate reason code to accounts selected for rerating. This enables you to rerate only accounts matching the reason for rerating.
- Set up automatic rerating, which automatically selects certain events for rerating so that you do not need to specify the accounts and events when you run the **pin_rerate** utility. You can set up the following automatic rerating features:
 - Automatic rerating for backdated events, rate changes, and rollover corrections
 - Out-of-order rerating for pipeline-rated events that are rated out of order (non-chronologically)
 - Trigger-dependent rerating for events based on custom rerating triggers that you configure. For example, you can automatically select events for rerating that are rated for a product that has been canceled.
- Create custom **pin_rerate** parameters, which enables you to select events based on any event criteria.

For more information about rerating by using **pin_rerate**, see "[About Comprehensive Rerating Using pin_rerate](#)".

About the Rerating Pipelines

Pipeline Manager rerates only the events that it previously rated. Pipeline Manager has two rerating pipelines. The one you use depends on whether you rerate events by using Pipeline Manager or by using **pin_rerate**:

- **The batch rerating pipeline.** This pipeline is used when rerating only pipeline-rated events by using Pipeline Manager. It receives and rerates events in

batches. The rerated events are loaded into the BRM database in batches by RE Loader.

- **The real-time rerating pipeline.** This pipeline is used when rerating pipeline-rated events by using `pin_rerate`. It rerates events individually that it receives from the CM. The rerated events are recorded into the BRM database as they are processed. See "[About Comprehensive Rerating Using pin_rerate](#)".

How BRM Applies the Balance Impacts of Rerating

BRM rerating handles both billed and unbilled events and rerating events across billing cycles. When events include non-currency balance impacts, BRM rerating properly redistributes non-currency resources and charges or credits the account accordingly. Financial impacts to accounts receivable (A/R), general ledger (G/L), and taxation are taken into account when adjustments are made as a result of rerating.

When an event is rerated, BRM backs out the event from the BRM database by creating an adjustment event that fully negates the original balance impacts. This adjustment event can be either a shadow adjustment event or a regular adjustment event:

- If the event is unbilled, a shadow adjustment event is created. See "[About Rerating Unbilled Events](#)".
- If the event has already been billed or posted to the G/L, a regular adjustment event is created. See "[About Rerating Billed and Posted Events](#)".

For rerated usage events, the balance impacts of rerating are applied by the adjustment event. For rerated cycle events, a new cycle event is created that applies the balance impacts of rerating. Cycle events include cycle fees, folds, rollovers, and cycle discounts.

When the total rerating adjustment is zero, BRM does not generate a rerating adjustment event if the balance impacts of rerating and previous rating are equivalent. To determine if the balance impacts are equivalent, BRM compares the values of certain balance impact fields for rerating with previous rating, such as the resource ID and balance group. If the values are different, BRM treats the rerated balance impacts as unique and generates rerating adjustment events.

You can customize how BRM determines whether the balance impacts of rerating and previous rating are equivalent by modifying the event balance impact fields that are used for comparison. For more information, see "[Determining Whether Balance Impacts of Rerating and Previous Rating Are Equivalent](#)".

How BRM Generates Rerated Events

BRM rerates events in chronological order:

- Usage events are rerated in order starting with the earliest usage event.
- For rollover, fold, and cycle discount events, BRM generates new events and applies the balance impacts according to the time that billing was run for the associated cycle.
- For cycle fee events, BRM generates new events and applies the balance impacts according to the time that the accounting cycle ends or when the product is purchased or canceled.

Rollover, fold, cycle discount, and cycle fee events are generated in the order in which they logically occur: for example, billing-time discount events are generated before rollover events.

Rerating generates events in the following ways:

- By rerating the original event
 This process passes the event being rerated to the rating opcode. Rerating generates an adjustment event to apply the balance impacts of rerating. This is the most basic rerating process.
- By reapplying business logic based on information in the original event
 This process passes information from the event being rerated to the opcode that generated the event. The opcode may or may not generate a new event. This process is used when the event attributes may be different after rerating. For example, a product's purchase fee may have been changed since the purchase fee event was generated. To reapply the purchase fee, rerating passes the product information in the original deal-purchase event to the PCM_OP_SUBSCRIPTION_PURCHASE_DEAL opcode, which generates a new purchase fee event, if applicable.

Note: In some cases, purchase and cancellation events are not available during rerating. In such cases, BRM rerates the purchase-fee or cancellation-fee event. For more information, see "[Events That Are Not Rerated](#)".

- By reapplying business logic independent of the original event
 This process performs business logic that may generate new events even if there is no existing event. With this process, information in the original event is not used. Instead, the business logic that generated the original event is reapplied and a new event is generated, if appropriate.

 For example, when rerating an account for a specific period, if the events selected for rerating are associated with products that have a cycle fee, rerating calls the cycle fee opcode and generates a new cycle fee event based on the product's current rates. This is done independent of any existing cycle fee event. If there is an existing cycle fee event, it is replaced by the corresponding new cycle fee event generated by rerating.

Table 23–1 shows the rerating process for specific event types.

Table 23–1 Rerating Process and Event Type

Event Type	Order of Rerating	How Rerated
Usage	Chronological, based on event time	Rerate the original event.
Cycle fee	Generated according to the time that the product is purchased or canceled or when the associated accounting cycle starts or ends, and based on the logical order of events	Reapply business logic independent of the original event.
Rollover	Generated according to the time that the associated billing cycle ends, and based on the logical order of events	Reapply business logic independent of the original event.
Fold	Generated according to the time that the associated billing cycle ends, and based on the logical order of events	Reapply business logic independent of the original event.

Table 23–1 (Cont.) Rerating Process and Event Type

Event Type	Order of Rerating	How Rerated
Billing-time discount	Generated according to the time that the associated billing cycle ends, and based on the logical order of events	Reapply business logic independent of the original event.
Purchase	Chronological, based on event time	Reapply business logic based on information in the original event.
Cancellation	Chronological, based on event time	Reapply business logic based on information in the original event.

About Rerating Unbilled Events

When unbilled events are rerated, a shadow adjustment event is created. This event is called a *shadow event* because its balance impact is added to the original event's bill item rather than to an adjustment item.

Rerating Unbilled Usage Events

When unbilled usage events are rerated, the shadow adjustment event fully negates the original balance impacts and applies new balance impacts for the rerated amount.

Rerating Unbilled Cycle Events

When unbilled cycle events are rerated, the shadow adjustment event fully negates the original balance impacts. Then a new cycle event is created that applies the rerated balance impacts. The new cycle event is of the same event type as the original event.

Note: When the purchase start time is the same as the current accounting cycle end date, the cycle forward fee balance impact is applied to the next month's bill instead of the current month's bill when billing is run after rerating is performed.

For example, suppose an account is created on January 1 with a cycle forward fee of \$20 and the purchase start time is deferred by 1 month (set to the accounting cycle end date). On February 1, when **pin_rerate** is run from January 1, because the purchase start time is the same as the accounting cycle end date, rerate internally triggers automatic billing. The billing process changes the status of the bill item to open. Because the bill item status is now open, rerating applies the rerate adjustments to the next bill. When regular billing is run on February 2, the cycle fee is not applied to the January bill; instead, it is applied to the February bill. For more information about rerating billed events, see "[About Rerating Billed and Posted Events](#)".

About Rerating Billed and Posted Events

When you rerate events that are already billed and unbilled events that are already posted to the G/L, the results of rerating are applied as adjustments to the next bill, which is the bill for the current cycle. The balance impacts are applied to the adjustment item.

Rerating Billed and Posted Usage Events

When usage events that are billed or posted to the G/L are rerated, an adjustment event fully negates the original balance impacts and applies new balance impacts for the rerated amount.

Rerating Billed and Posted Cycle Events

For billed cycle fee, fold, rollover, and cycle discount events, an adjustment event is created that fully negates the original balance impacts. The adjustment is applied to the current billing cycle and does not impact the original event's items. A new cycle event is then created that applies the rerated balance impacts. The new cycle event is of the same event type as the original event and is applied to the current cycle.

When billing is run, if a non-currency resource has a zero balance, no cycle event is generated for that resource. For example, if there are no remaining free minutes to roll over at billing time, no rollover event is created. However, if rerating creates a nonzero amount for a resource that was previously zero when billed, the rerating process generates the appropriate cycle event.

About Rerating and Pricing Changes

Changes to account products are audited. If you change a product attribute after events are rated, and then rerate those events, the rerating process uses only the current account subscription data.

Events That Are Not Rerated

Usually, purchase-fee events (`/event/billing/product/fee/purchase`) and cancellation-fee events (`/event/billing/product/fee/cancel`) are not directly rerated. Instead, information in the original deal purchase event (`/event/billing/deal/purchase`) and product cancellation event (`/event/billing/product/action/cancel`) is resent to the `PCM_OP_SUBSCRIPTION_PURCHASE_DEAL` and `PCM_OP_SUBSCRIPTION_CANCEL_PRODUCT` opcodes respectively. The opcodes generate new fee events, if applicable.

In some cases, the purchase-fee and cancellation-fee events are directly rerated because the purchase and cancellation events may not be available for rerating; for example, when rerating deferred purchase fees or when using selective rerating in which only purchase-fee or cancellation-fee events are specified.

The following events are not rerated but are reapplied because they were not originally generated by rating:

- `/event/billing/adjustment`
- `/event/billing/charge`
- `/event/billing/cycle/tax`
- `/event/billing/debit`
- `/event/billing/dispute`
- `/event/billing/item`
- `/event/billing/payment`
- `/event/billing/refund`
- `/event/billing/reversal`
- `/event/billing/settlement`

- [/event/billing/writeoff](#)

About Rerating Events for Account Sharing Groups

When rerating accounts and bill units in a hierarchy, sponsor, or discount sharing group, BRM does not automatically rerate subordinate and member accounts and bill units. To rerate subordinate and member accounts and bill units, you must specify them when you select the accounts for rerating. However, when BRM rerates an event for a service associated with sponsorship, rerating impacts the appropriate sponsoring or sponsored account's balance.

About Rerating Events That Impact Sponsored Resources

If the balance impacts of an event are sponsored by another account, the balance impacts of rerating are applied to the sponsoring account. For example, account A pays for Internet access for account B. If account B's events are rerated, the balance impacts of rerating the Internet usage events are applied to account A's balance.

Likewise, if a rerated event impacts a resource that is shared with another account, the resource in the sponsored account is adjusted. For example, account A gets 1 frequent flyer mile for every dollar that account B spends. If account B's events are rerated and its current balance is reduced from \$100 to \$50, account A's frequent flyer miles are adjusted from 100 to 50.

If a rerated, sponsored event impacts a resource that is shared by more than two accounts, rerating adjusts only the balances in the rerated account and the sponsoring account. For example, account A shares free minutes with accounts B and C. If account B and account C both make calls, and then account B is rerated, any impact on the free minutes account B used is applied to account A. However, that free-minute balance impact in account A is not carried over to account C, even though account C shares those resources.

BRM Functionality Affected by Rerating

When you rerate events, the following areas in BRM are affected:

- General ledger entries. See "[Determining the G/L Entry for an Event](#)".
- Invoices. See "[Displaying Shadow-Event Rerating Details on Invoices](#)".

Determining the G/L Entry for an Event

The G/L entry is determined at the time of rating. If you rerate as a result of pricing changes, the G/L entry could change. Whether you record a shadow event or an adjustment event, you must determine the correct balance to be posted in the G/L.

When recording a shadow event, the G/L ID of the rerating balance impacts have the same G/L ID as the original event's balance impacts. If you changed the G/L ID after the original event was rated, the balance impacts of rerating use the new G/L ID.

When recording an adjustment, you can configure the G/L ID to use. For example, you can use the same G/L ID as the original event, a new G/L ID, or an adjustment G/L ID (a separate G/L ID bucket to record all adjustments as part of rerating). The adjustment G/L ID can be the same or different than the G/L ID used for regular (not rerated) event adjustments. For information about setting up G/L IDs, see "About Collecting General Ledger Data" in *BRM Collecting General Ledger Data*.

Displaying Shadow-Event Rerating Details on Invoices

When recording a shadow event, you can customize your invoice to either show the details of rerating or show the result of rerating only in an end balance.

To customize your invoice:

1. Open the *BRM_Home/sys/cm/pin.conf* file.
2. Set the **show_rerate_details** entry:
 - To show the details of rerating, set the entry to **1**.
 - To show the result of rerating only in an end balance, set the entry to **0**. The default is **0**.

For example:

```
- fm_inv_pol show_rerate_details 0
```

3. Save and close the file.
4. Stop and restart the CM. For more information, see "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.

Determining Whether Balance Impacts of Rerating and Previous Rating Are Equivalent

When usage events are rerated, an adjustment event is generated to apply the balance impacts of rerating. The total balance impacts of the adjustment event is the difference between the previous rating and rerating results.

When rerating produces the same result as previous rating, creating an adjustment event is not necessary. However, because an event can include multiple balance impacts, BRM must ensure that the balance impacts in each rating process are actually equivalent.

To determine whether the balance impacts of rerating and previous rating are equivalent, BRM checks the following values of the balance impacts in the event. If any of these values is different for rerating than it is for previous rating, BRM treats the associated balance impact as being unique and generates an adjustment event that includes the balance impact:

- The ID of the resource being impacted
- The ID of the balance group being impacted
- The G/L ID
- The tax code
- The product used to rate the event

The list of default balance impact fields used to compare the balance impacts of events' rerating and previous rating results is stored in the **/config/rerate_flds/compare_bi** object in the BRM database. You can modify the balance impact fields stored in the **/config/rerate_flds/compare_bi** object to customize how BRM determines whether balance impacts are equivalent.

For example, if you remove the product field and the total rerating adjustment is zero, adjustment events are not generated even when the product used to rerate events is different than the product previously used (providing the other balance impact fields are the same). Or, if you specify additional balance impact fields, adjustment events are generated when the values of those fields differ. For example, you can specify the

impact category field to record the rerating adjustment when calls are made from different locations even though rerating results in the same total charge.

To customize the list of event balance impact fields that determine whether rerated and previously rated balance impacts are equivalent, see ["Specifying How to Compare Balance Impacts When Creating Adjustment Events"](#).

Specifying How to Compare Balance Impacts When Creating Adjustment Events

BRM checks the `/config/erate_flds/compare_bi` object during rerating. If the balance impact fields in this object have the same values in the event for both rerating and previous rating, and the total rerating adjustment is zero, the results of rerating and previous rating are considered equivalent and no adjustment event is created.

Caution: The `load_pin_erate_flds` utility overwrites existing balance-impact comparison fields. If you are updating balance-impact comparison fields, you cannot load new fields only: you must load complete sets of the balance-impact comparison fields when you run the `load_pin_erate_flds` utility.

Important: The `load_pin_erate_flds` utility uses a configuration file (`pin.conf`) located in the same directory to connect to the BRM database. Edit the configuration file to connect to your BRM database.

Note: The `load_pin_erate_flds` utility loads extraction keys that define custom `pin_erate` parameters (see ["Defining Custom pin_erate Parameters for Rerating"](#)). You cannot use the same file to specify extraction keys and balance-impact comparison fields; you must run `load_pin_erate_flds` with separate files for each configuration.

To customize the fields used to compare the event balance impacts of rerating with previous rating:

1. Open the `BRM_Home/sys/data/config/pin_erate_compare_bi.xml` file.

Add a **RerateCompareBalImpacts** element for each event balance impact field. The following balance impact fields are mandatory:

- PIN_FLD_RESOURCE_ID
- PIN_FLD_BAL_GRP_OBJ
- PIN_FLD_GL_ID
- PIN_FLD_TAX_CODE

Note: `PIN_FLD_PRODUCT_OBJ` is specified in the `pin_erate_compare_bi.xml` file by default, but it is an optional field.

You can specify any additional field from the `/event` object's `PIN_FLD_BAL_IMPACTS` array.

2. Save and close the file.

3. Run the following command, which loads the contents of the XML file into the `/config/rerate_flds/compare_bi` object:

```
load_pin_rerate_flds pin_rerate_compare_bi.xml
```

If you do not run the utility from the directory in which the XML file is located, you must include the complete path to the file. For example:

```
load_pin_rerate_flds BRM_Home/sys/data/config/pin_rerate_compare_bi.xml
```

For more information, see "[load_pin_rerate_flds](#)".

4. Stop and restart the CM. For more information, see "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.
5. To verify that the balance-impact comparison fields were loaded, display the `/config/rerate_flds/compare_bi` object by using Object Browser, or use the `robj` command with the `testnap` utility.

How BRM Tracks Rerated Events

This section is useful if you write custom code that requires accessing events that were rerated.

To back out original rated events, BRM generates adjustment events (`/event/billing/adjustment/event`). These events contain a compliment of all balance impacts in the original events to fully negate the events. When an event is rerated, the `PIN_FLD_RERATE_OBJ` field in the original rated event references this backout adjustment event.

When usage events are rerated, the new balance impacts of rerating are included in the same adjustment event that backs out the original balance impacts. This means the original event references the new event that contains the rerated balance impacts.

When cycle fee, cycle discount, fold, and rollover events are rerated, new cycle events are created to apply the balance impacts of rerating. In this case, there is no relationship between the original events and the new (cycle) events that contains the rerated balance impacts.

Because you can use flexible cycles and multiple discounts, there might be more than one rollover, fold, and discount event to rerate for a given billing cycle:

- A fold event is generated per balance for each resource that has a fold.
- A rollover event is generated per balance group for each product that has a rollover.
- A discount event is generated for each cycle discount.

Configuring BRM for Elastic Charging Engine Rerating

You can rerate events using Oracle Communications Elastic Charging Engine (ECE).

To configure BRM for ECE rerating:

1. Enable and load the `ECERating` business parameter. See "[Enabling Elastic Charging Engine Rerating](#)".
2. Configure the CM for rerating. See "[Configuring the Connection Manager for Elastic Charging Engine Rerating](#)".

3. Verify that the ECE `payloadconfig_ece_sync.xml`, `pin_notify`, `event_t.ctl`, and `event_dlay_sess_tlcs_t.ctl` configuration files are loaded into the BRM environment. See "[Loading the Elastic Charging Engine Configuration Files into Your BRM Environment](#)".
4. Verify that the acknowledgment queue (Oracle AQ database queue) is created.

Note: The acknowledgment queue is created by an ECE post-installation script. For more information, see the discussion on ECE post-installation tasks in *ECE Installation Guide*.

5. Verify that the ECE External Manager (EM) Gateway is running. For more information, see the discussion on starting and stopping ECE in *ECE System Administrator's Guide*.

You can run rerating using the `pin_rerate` utility. For more information on how to run rerating using the `pin_rerate` utility, see "[pin_rerate](#)".

Enabling Elastic Charging Engine Rerating

By default, the `ECERating` business parameter is disabled in BRM.

To enable ECE rerating:

1. Go to the `BRM_Home/sys/data/config` directory.
2. Run the following command, which creates an editable XML file from the `rerate` instance of the `/config/business_params` object:

```
pin_bus_params -r BusParamsRerate bus_params_rerate.xml
```

This command creates the XML file named `bus_params_rerate.xml.out` in your working directory. To place this file in a different directory, specify the path as part of the file name.

3. Open the `bus_params_rerate.xml.out` file.
4. Search for the following line:


```
<ECERating>disabled</ECERating>
```
5. Change `disabled` to `enabled`.
6. Save the file as `bus_params_rerate.xml`.
7. Go to the `BRM_Home/sys/data/config` directory, which includes support files used by the `pin_bus_params` utility.
8. Run the following command, which loads this change into the `/config/business_params` object:

```
pin_bus_params PathToWorkingDirectory/bus_params_rerate.xml
```

where `PathToWorkingDirectory` is the directory in which the `bus_params_rerate.xml` file resides.

Caution: BRM uses the XML in this file to overwrite the existing `rerate` 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 `rerate` configuration.

Note: To run this command from a different directory, see "pin_bus_params" in *BRM Developer's Guide*.

9. Read the object with the **testnap** utility or Object Browser to verify that all fields are correct.

See "Using testnap" in *BRM Developer's Guide* for general instructions on using the **testnap** utility. See "Reading Objects by Using Object Browser" in *BRM Developer's Guide* for information on how to use Object Browser.
10. Stop and restart the CM. For more information, see "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.
11. (Multischema systems only) Run the **pin_multidb** script with the **-R CONFIG** parameter. For more information, see "pin_multidb" in *BRM System Administrator's Guide*.

Configuring the Connection Manager for Elastic Charging Engine Rerating

BRM CM connects to ECE through ECE EM Gateway. For rerating to work with ECE, the ECE EM Gateway group and pointer entries must be configured in the CM.

To configure the CM for ECE rerating:

1. Open the *BRM_Home/sys/cm/pin.conf* file in a text editor.
2. For real-time rerating, uncomment the following entry:

```
- cm fm_module BRM_home/lib/fm_rerate.so fm_rerate_config - pin
```
3. Search for the following line:

```
- cm em_group rerating PCM_OP_RATE_PIPELINE_EVENT
```
4. Replace it with the following:

```
-cm em_group rerating PCM_OP_RATE_ECE_EVENT
```
5. Set the rerating **em_pointer** entry to match your environment:

```
- cm em_pointer rerating ip emGateway_host emGateway_port
```

where:

- *emGateway_host* is the IP address of the machine on which the ECE EM Gateway resides.
 - *emGateway_port* is the port on which the ECE EM Gateway is running.
6. Save and close the file.
 7. Stop and restart the CM. For more information, see "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.

Loading the Elastic Charging Engine Configuration Files into Your BRM Environment

The ECE BRM Integration Pack contains the following configuration files that you must load into your BRM environment:

- *ECE_Home/occeserver/brm_config/payloadconfig_ece_sync.xml*
See "[Loading the ECE Payload Configuration File](#)".

- *ECE_Home/occeserver/brm_config/pin_notify*
See "[Loading the ECE Event Notification List](#)".
- *ECE_Home/occeserver/brm_config/event_t.ctl*
- *ECE_Home/occeserver/brm_config/event_dlay_sess_tlcs_t.ctl*
See "[Updating the RE Loader Control Files with ECE Control Data](#)".

where *ECE_Home* is the directory in which ECE is installed.

Loading the ECE Payload Configuration File

The ECE payload configuration file (*payloadconfig_ece_sync.xml*) defines business events that include BRM data needed by ECE.

To load the ECE payload configuration file into your BRM environment:

1. Copy the *ECE_Home/occeserver/brm_config/payloadconfig_ece_sync.xml* file to the *BRM_Home/sys/eai_js* directory.
2. Do one of the following:
 - If your BRM system does not already have an EAI publisher:
 - a. Open the *BRM_Home/sys/eai_js/Infranet_eai.properties* file in a text editor.
 - b. Set the **infranet.eai.configFile** entry to point to the ECE payload configuration file:


```
infranet.eai.configFile=./payloadconfig_ece_sync.xml
```
 - c. Save and close the file.
 - If your BRM system already has an EAI publisher:
 - a. Merge the *payloadconfig_ece_sync.xml* file with the existing payload configuration file.
 - b. Set the **infranet.eai.configFile** entry in the *BRM_Home/sys/eai_js/Infranet_eai.properties* file to point to the merged file.

See "Configuring the EAI Payload for Account Synchronization" in *BRM Installation Guide*.

3. Go to the *BRM_Home/bin* directory.
4. Stop and restart the Payload Generator External Module (the EAI Java server) by running the following command:

```
pin_ctl restart eai_js
```

Loading the ECE Event Notification List

The ECE **pin_notify** configuration file contains an event notification list that BRM uses to send update requests to ECE.

To load the ECE event notification list into your BRM environment:

1. Open your current BRM event notification file and the *ECE_Home/occeserver/brm_config/pin_notify* file in a text editor.

Tip: Save a copy of the original files before proceeding.

2. Merge the files by copying the contents of the *ECE_Home/occeserver/brm_config/pin_notify* file to the end of the BRM event notification file. See "Merging Event Notification Lists" in *BRM Developer's Guide*.
3. Save and close the BRM event notification file.
4. Close the *ECE_Home/occeserver/brm_config/pin_notify* file.
5. Load the merged file into the database by running the following command:

```
load_pin_notify -v event_notification_configuration_file_name
```
6. Stop and restart the CM. For more information, see "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.

Updating the RE Loader Control Files with ECE Control Data

The following ECE control files are used by RE Loader. RE Loader uses these files for processing rated events in CDRs coming from ECE.

- **event_t.ctl**: Includes control data used for loading data from the ECE usage request into columns of the EVENT_T table.
- **event_dlay_ses_tlcs_t.ctl**: Includes control data used for loading input and output volume data from the ECE usage request into columns of the EVENT_DLAY_SES_TLCS_T table.

To update the RE Loader control files with ECE control data:

1. Go to *BRM_Home/apps/pin_rel* directory.
2. Copy the *ECE_Home/occeserver/brm_config/event_t.ctl* and *ECE_Home/occeserver/brm_config/event_dlay_ses_tlcs_t.ctl* control files to each processing directory in *BRM_Home/apps/pin_rel*.
 For example, if you have a processing directory called **GPRS**, copy the ECE **event_t.ctl** and ECE **event_dlay_ses_tlcs_t.ctl** control files to *BRM_Home/apps/pin_rel/GPRS*.

See "Setting Up RE Loader Processing Directories" in *BRM Configuring Pipeline Rating and Discounting*.

3. If you have customized control files, modify ECE control files to load your custom data.

Customized control files are used, for example, when you write an extension mapping of an ECE usage request attribute in RE Loader.

For more information, see "Adding Support for a New Service" in *BRM Developer's Guide* and "Adding New Event Types for RE Loader to Load" in *BRM Configuring Pipeline Rating and Discounting*.

For information about adding custom ECE usage-request attributes, see the discussion on creating new usage requests in *ECE Implementation Guide*.

About Real-Time Rerating Pipelines

This chapter describes:

- How events are processed by Oracle Communications Billing and Revenue Management (BRM) real-time rerating pipelines.
- How to configure real-time rerating pipelines.

Before reading this chapter, you should be familiar with the Pipeline Manager architecture. See *BRM Configuring Pipeline Rating and Discounting*.

About Real-Time Rerating Pipelines

When you run the "[pin_rerate](#)" utility to rerate events rated in real-time and events rated in a batch pipeline, you run the **pin_rerate** utility to send pipeline-rated events to a real-time rerating pipeline to be rerated. See "[About Comprehensive Rerating Using pin_rerate](#)".

The Connection Manager (CM) sends pipeline-rated events in the form of a flist to the NET_EM pipeline module. The NET_EM module transfers the event to a real-time rerating pipeline for rerating.

The real-time rating opcodes add to the flist all the enrichment data needed by the real-time rerating pipeline to rate the event. For example, if the real-time rating opcodes determine that a discount should be applied, it adds the discount information to the flist.

Similar to a batch rerating pipeline, a real-time rerating pipeline performs both rating and discounting functions.

To rerate events, the real-time rerating pipeline gets the new pricing information from the Pipeline Manager database. Certain configuration data, such as currency and non-currency resource information, are obtained from the BRM database.

Overview of Event Processing in the Real-Time Rerating Pipeline

1. The INP_Realttime input module converts the flist received from the NET_EM module to event data record (EDR) format and creates an EDR container.

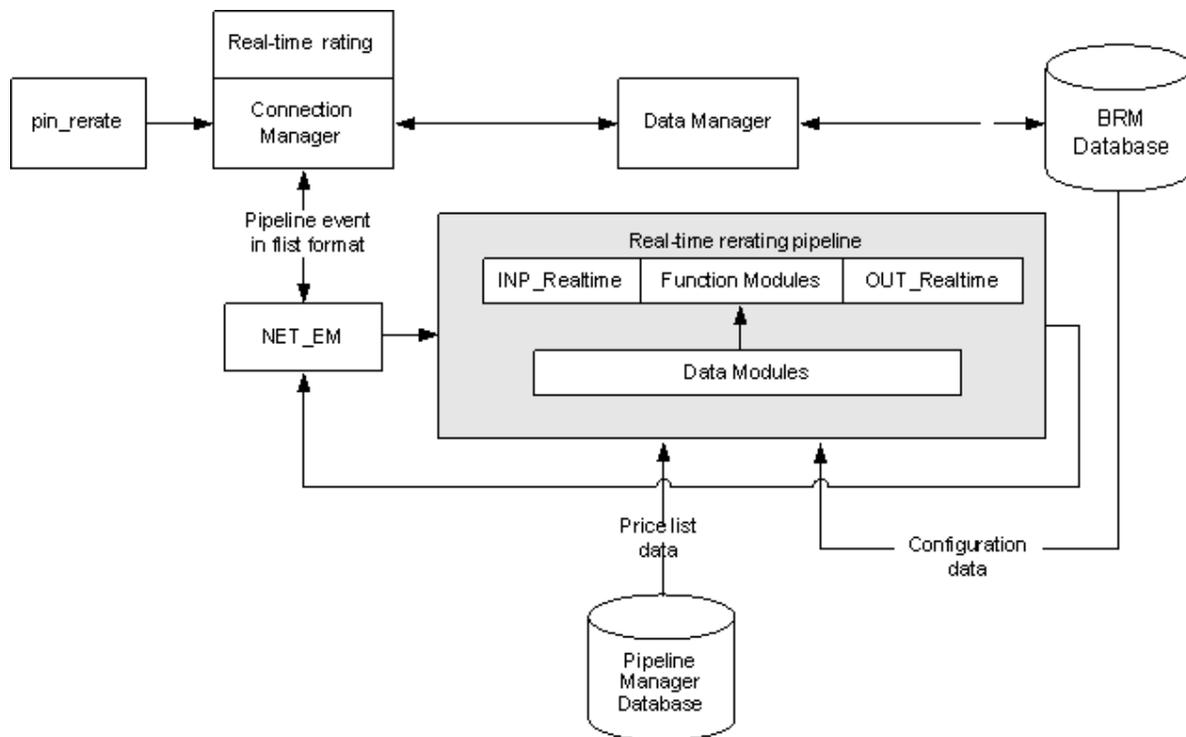
Tip: You can also create and run an iScript to manipulate data in the EDR container before it is sent to the pipeline modules.

2. The pipeline function modules calculate the new balance impacts by using the new pricing and discounting information and then add the balance impacts to the EDR container.
3. The OUT_Realttime module converts the EDR back into flist format and sends the flist to NET_EM.
4. NET_EM sends the flist with the new balance impacts back to the CM.
5. BRM updates the account's balances and records the event in the BRM database.

For detailed information about configuring the real-time rerating pipeline, see ["Configuring Rerating of Pipeline-Rated Events in the Real-Time Rerating Pipeline"](#).

Figure 24–1 shows the data flow for rerating pipeline-rated events by using the real-time rerating pipeline:

Figure 24–1 Rerating Data Flow for Pipeline-Rated Events



About Transaction Management for the Real-Time Rerating Pipeline

The real-time rerating pipeline does not use the Transaction Manager (TAM) module. Instead, transaction handling is provided by the CM. When a rerating transaction fails, the CM rolls back the transaction and restores all the account balances in the BRM database. For this reason, function modules and iScripts used by the real-time rerating pipeline are stateless.

Important: If you use iScripts for custom processing, ensure that the iScripts are stateless.

Configuring Rerating of Pipeline-Rated Events in the Real-Time Rerating Pipeline

To configure rerating of pipeline-rated events in the real-time rerating pipeline, perform the following tasks:

- Configure the real-time rerating pipeline. See "[Configuring a Real-Time Rerating Pipeline](#)".

Important: When you configure pipelines in the **Pipelines** section, you must add the pipelines to the IFW_PIPELINE table. Otherwise, you will receive an error at system startup. For more information, see "[Configuring the Real-Time Rerating Pipelines in the IFW_PIPELINE Table](#)".

- Configure the CM to send rerate requests to the NET_EM module.
- Configure NET_EM to route rerate requests to the real-time rerating pipeline.

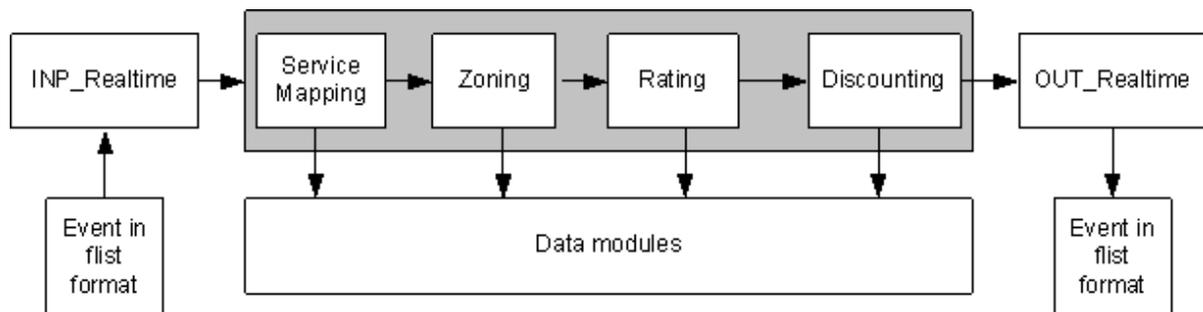
Configuring a Real-Time Rerating Pipeline

A real-time rerating pipeline is configured similarly to a batch rerating pipeline, but with fewer preprocessing and post-rating modules. This is because most of the enrichment data is provided in the input flist received from the CM.

You typically create a separate instance of Pipeline Manager for real-time rerating. The default registry file is *Pipeline_home/conf/wirelessRealtime.reg*.

Figure 24–2 shows the real-time rerating pipeline architecture:

Figure 24–2 Real-Time Rerating Pipeline Architecture



Configuring Multiple Real-Time Rerating Pipelines

To improve performance or scalability, you configure multiple instances of real-time rerating pipelines. For example, you can increase performance by configuring multiple pipelines to process rerate requests in parallel or by configuring multiple real-time rerating pipelines to process different rerate requests: for example, by configuring one pipeline that rerates only GPRS events and another that rerates only GSM events.

Configuring the Real-Time Rerating Data Pool

Configure the following data modules:

- DAT_Zone

- Rateplan
- DAT_Calendar
- DAT_TimeModel
- DAT_PriceModel
- Dayrate
- DAT_Currency
- DAT_USC_Map

See "Configuring the data pool" in *BRM System Administrator's Guide*.

Configuring the Modules in the Real-Time Rerating Pipeline

To configure the modules in the real-time rerating pipelines, perform the following tasks:

- Configure the input module to use the XML file containing the flist-to-EDR mappings.
Use this entry for the **OpcodeMapping** entry:

```
OpcodeMapping = ../formatDesc/Formats/Realtime/rate_event.xml
```
- Configure NET_EM to manage data between the CM and Pipeline Manager.
- Configure the output module to add any customizations to the output flist.
- Configure these function modules to perform rerating:
 - FCT_ServiceCodeMap
 - FCT_CustomerRating
 - FCT_PreRating
 - FCT_IRules
 - FCT_USC_Map
 - FCT_RSC_Map
 - FCT_MainRating
 - FCT_Dayrate
 - FCT_RateAdjust
 - FCT_Rounding
 - FCT_DiscountAnalysis
 - FCT_Discount

Configuring the Real-Time Rerating Pipeline to Set Product Validity Periods

If your products are configured to start when they are first used, configure the ISC_FirstProductRealtime iScript in the real-time rerating pipeline.

For information about first-usage start times, see 12io12,lw-023p, and "[About Balance Impacts That Become Valid on First Usage](#)".

When products start on first usage, the real-time rerating pipeline adds the account's first-usage product and discount information to the EDR. ISC_FirstProductRealtime

sets the validity period in the BRM database for those products that were used to rate the event and that start on first usage. This triggers any purchase and cycle fees.

Configuring the Real-Time Rerating Pipelines in the IFW_PIPELINE Table

Pipeline Manager stores information about pipelines in the IFW_PIPELINE table. The pipelines that are preconfigured in the **Pipelines** section of the default registry file (*Pipeline_home/conf/wirelessRealtime.reg*) are inserted into the IFW_PIPELINE table during Pipeline Manager installation.

Important:

- If you are *not* using the default registry file, and you have configured new real-time rerating pipelines, you must manually insert the pipelines into the IFW_PIPELINE table by using SQL commands. Otherwise, you will receive an error at system startup.
 - If you *are* using the default registry file and have changed the default pipeline names or you have configured additional pipelines, you must manually insert the new pipelines into the IFW_PIPELINE table.
-
-

The following example shows SQL commands to insert RealtimeReratingGSM and RealtimeReratingGPRS pipelines into the IFW_PIPELINE table:

```
% sqlplus pin/password@databaseAlias

SQL>INSERT INTO IFW_PIPELINE ( PIPELINE, NAME, EDRC_DESC ) VALUES (
'RealtimeReratingGSM', 'GSM Realtime Rerating Pipeline', 'ALL_RATE');

SQL>INSERT INTO IFW_PIPELINE ( PIPELINE, NAME, EDRC_DESC ) VALUES (
'RealtimeReratingGPRS', 'GPRS Realtime Rerating Pipeline', 'ALL_RATE');

SQL>commit;
```

Configuring NET_EM to Route Rerate Requests Based on the Event Field Value

To configure NET_EM to route rerate requests to multiple real-time rerating pipelines based on the type of event, you set the **FieldName** and **FieldValue** NET_EM module registry entries.

Important: If you use event routing based on the event field value, ensure that the input events contain the expected field name and field values specified in the NET_EM module registry. Otherwise, NET_EM will not be able to route the events.

By using the “.” notation, you can specify a field at any level in the input event list to be used to route the event. For example, this substruct and field:

```
PIN_FLD_EVENT
  PIN_FLD_POID
```

is represented like this:

```
PIN_FLD_EVENT.PIN_FLD_POID
```

In the NET_EM registry below, if PIN_FLD_EVENT.PIN_FLD_POID is a GSM event, the rerate request is routed to any one of the two instances of the GSM rerating pipeline (RealtimeReratingGSM). If the event is a GPRS event, the rerate request is routed to any one of the two instances of the GPRS rerating pipeline (RealtimeReratingGPRS).

```
DataPool
{
  RealtimePipeline
  {
    ModuleName = NET_EM
    Module
    {
      ThreadPool
      {
        Port = 14579
        Threads = 4
      }

      ReratingOpcode
      {
        OpcodeName = PCM_OP_RATE_PIPELINE_EVENT
        FieldName = PIN_FLD_EVENT.PIN_FLD_POID
        GSMEvent
        {
          FieldValue = /event/delayed/session/telco/gsm
          PipelineName = RealtimeReratingGSM
          NumberOfRTPipelines = 2
        }
        GPRSEvent
        {
          FieldValue = /event/delayed/session/gprs
          PipelineName = RealtimeReratingGPRS
          NumberOfRTPipelines = 2
        }
      }
    }
  }
}
```

About Rerating Pipeline-Rated Events

This chapter provides an overview of how you can rerate events that were originally rated in batch by Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager.

For information about rerating events rated in real time, see "[About Comprehensive Rerating Using pin_rerate](#)".

Overview of Rerating Events

It is possible to discover pricing or rating configuration errors after events have been rated. For example, a sample review of invoices might reveal a price configuration error. In addition to correcting the price configuration, you might need to rerate events that were rated with the incorrect configuration.

Use the following components to rerate events that were originally rated by Pipeline Manager:

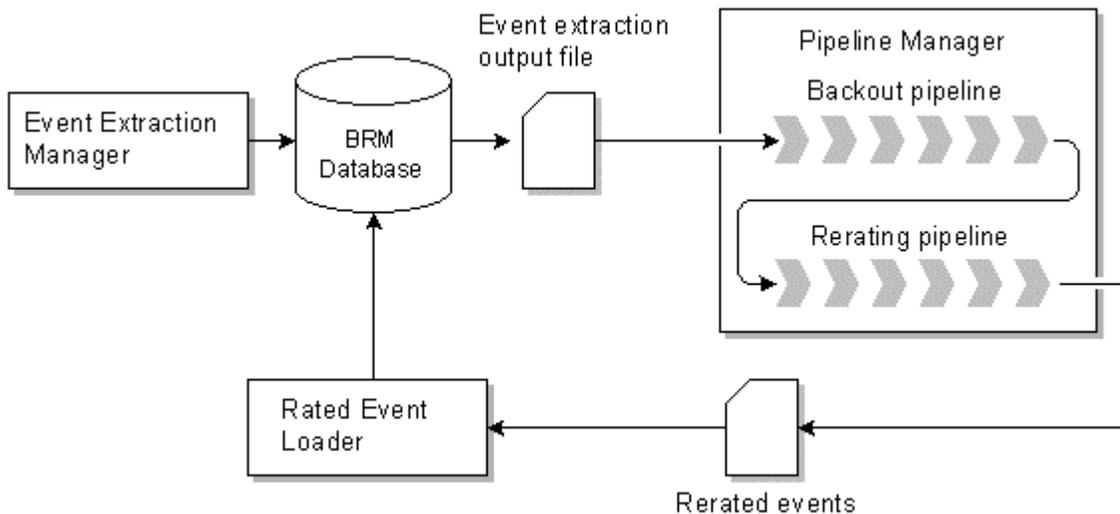
1. Use the *Event Extraction utility* to obtain data from the BRM database about the events you must rerate. This utility writes event data to a file.
2. Use *Pipeline Manager* to process the file that was created by the Event Extraction utility. There are two steps:
 - a. Restore discount and aggregation data to the state it was in before rating the events. For example, if you use volume discounting based on calls made, you can adjust the number of calls made.
 - b. Rerate the events using the new rating data. This process creates an output file.
3. Use the *Rated Event Loader (RE Loader)* to load the output file generated by Pipeline Manager into the BRM database.

Pipeline Manager only rerates events that it has previously processed. When you use pipeline rerating, there might be events that are not rerated. For example, if you retrieve events for rerating based on the account, any events belonging to the account that were generated in BRM and not previously processed by Pipeline Manager are not rerated. Ensure that all events are rerated by also running the BRM rerating utility (`pin_rerate`). See "[About Comprehensive Rerating Using pin_rerate](#)".

Important: When you rerate wireless events by using Pipeline Manager, you rerate only EDRs that have been rated by Pipeline Manager. Events that have been rated in real time are rerated by using the `pin_rerate` utility.

Figure 25–1 shows the rerating process:

Figure 25–1 Rerating Process



About Extracting Events from the BRM Database

The Event Extraction Manager searches for and extracts events meeting a specified search criteria and writes the results to an output file. It does not modify or process any of the events, nor does it lock accounts in your database. This enables BRM and other applications to safely access accounts in the database while you are extracting events.

To extract events, you perform the following steps:

1. Edit the event search file to specify which events to extract. You can specify events based on many different event attributes, for example, date, account, product, rate plan, and service. See "[Event Search Criteria](#)".
2. Configure and run the `pin_event_extract` utility to create the event extract output file. This file contains the list of events to rerate in EDR (event data record) format, and is used as the input for the Pipeline Manager rerating process.

For more information about extracting events for rerating, see "[Using Event Extraction Manager](#)".

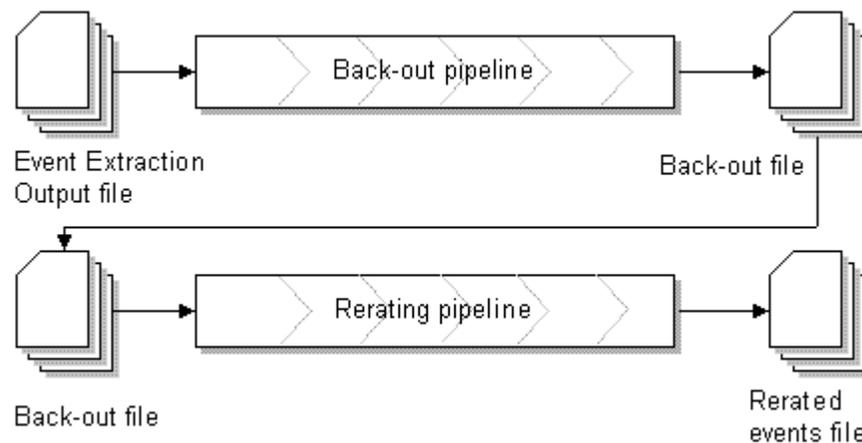
About Rerating Events with Pipeline Manager

To rerate events with Pipeline Manager, you perform the following steps:

1. Correct the rating configuration that caused the non-valid rating.
2. Configure the back-out pipeline. This pipeline uses the event extract output file as input. It backs out the existing balance impacts from discount accounts, aggregates the results, and outputs a back-out file.
3. Configure the rerating pipeline. This pipeline uses the back-out file as input. It rerates the events using the corrected rating configuration. The rerating process is the same as the rating process, however, the output file also includes the delta of the old and new balance impacts. Rated Event Loader loads the file into the BRM database.

Figure 25–2 shows the rerating steps described.

Figure 25–2 Pipeline Manager Rerating Process



For more information, see "[Configuring Rerating in Pipeline Manager](#)".

About Loading Rerated Events into the BRM Database

You load rerated events into the BRM database by running the Rated Event Loader (RE Loader). RE Loader identifies events that have already been billed, ensuring that account balances are accurately updated. See "[Adjusting Account Balances](#)".

To load rerated events into the BRM database, you perform the following steps:

1. Set up the RE Loader processing directories.
2. Configure RE Loader to load the rerated event types by configuring the `RELoader_home/Infranet.properties` file.
3. Run RE Loader.

For more information about RE Loader, see "Understanding Rated Event Loader" in *BRM Configuring Pipeline Rating and Discounting*.

Adjusting Account Balances

RE Loader determines if each event has been billed. For events that *have* been billed, RE Loader creates an adjustment event (`/item/adjustment`) with the appropriate balance impact and sets the event type to `/event/billing/adjustment/event`. It then loads the events and updates the account balances. RE Loader updates the event as having been rerated and keeps a history of each update.

About Back-Out-Only Rerating

You use back-out-only rerating to reprocess events that were originally rated by Pipeline Manager. It enables you to back out only the event balance impacts without rerating the events.

For example, you might want to use back-out-only rerating when you find that call detail records (CDRs) rated and recorded in the BRM database contain incorrect values. In such cases, you can use back-out-only rerating to back out the event balance impacts without rerating the event.

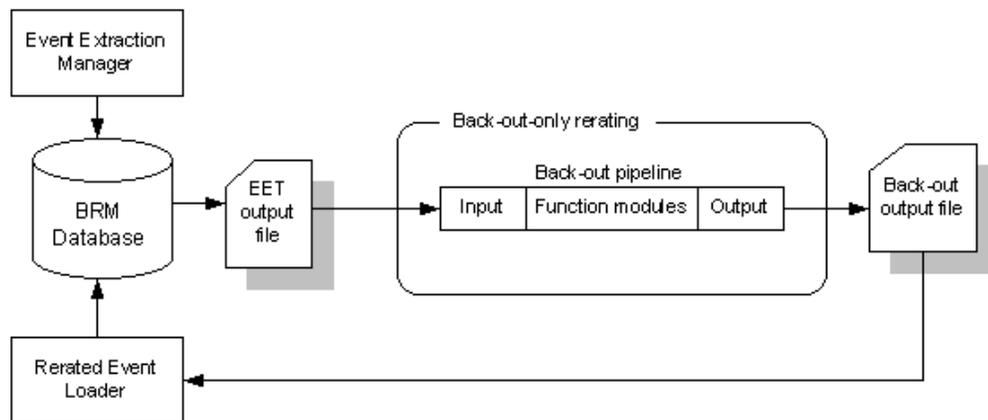
The output file created by backout-only rerating is loaded into the BRM database by the Rated Event (RE) Loader.

To process events for back-out-only rerating:

1. Use the Event Extraction Manager to extract the events from the BRM database. See "[About Extracting Events for Back-Out-Only Rerating](#)".
2. Use the backout pipeline to process the output file generated by the Event Extraction Manager. See "[About Configuring the Backout Pipeline for Back-Out-Only Rerating](#)".
3. Use the Rated Event Loader to load the output file generated by the backout pipeline into the BRM database. See "[About Loading Rated Events into the BRM Database](#)".

Figure 25–3 shows the back-out-only rerating process:

Figure 25–3 Back-Out-Only Rerating Process



About Extracting Events for Back-Out-Only Rerating

Use the Event Extraction Manager to find and extract events from the BRM database for backout-only processing. The Event Extract Tool (`pin_event_extract`) generates an output file containing the events in EDR format.

To extract events for back-out-only rerating, run `pin_event_extract` with the `-e` parameter.

Important: If you do not use the `-e` parameter, Pipeline Manager sends the EDRs in the output file to the rerating pipeline for rerating.

For detailed information about specifying the search criteria and running the `pin_event_extract` utility, see "[Using Event Extraction Manager](#)".

About Configuring the Backout Pipeline for Back-Out-Only Rerating

You use the backout pipeline to process the EDRs extracted by the Event Extraction Manager. The backout pipeline backs out the event balance impacts and writes the events to an output file.

The backout pipeline generates output files for both normal rerating and back-out-only rerating:

- In normal rerating, the output file is processed by the batch-rerating pipeline.
- In back-out-only rerating, the output file is processed by RE Loader.

Both types of rerating require different file formats and output grammar.

RE Loader loads events of different types from separate directories. For this reason, with back-out-only rerating, the backout pipeline sends the EDRs to different output streams based on the event type. For example, the backout pipeline writes all GSM events in one file, and all GPRS events in another file, and sends them to separate output directories.

To configure back-out-only rerating:

1. Configure the `OUT_GenericStream` module to generate the output file using the output grammar for back-out-only rerating and to send the output files to different directories based on the event type. See "[Configuring the OUT_GenericStream Module](#)".
2. Configure RE Loader to load the back-out output file into the BRM database. See "[About Loading Rerated Events into the BRM Database](#)".

Configuring the `OUT_GenericStream` Module

The `OUT_GenericStream` module formats the output for back-out-only rerating based on the grammar file specified in its registry. The `EXT_OutFileManager` sends the data to the output directories based on the event type specified in its registry. You configure the `EXT_OutFileManager` module inside the `OUT_GenericStream` module.

To configure the `OUT_GenericStream` and `EXT_OutFileManager` modules:

- Set the **Grammar** entry to


```
./formatDesc/Formats/Solution42/V670_EVENT_LOADER_OutGrammar.dsc.
```
- Set the **OutputPath** entry to specify the output directory. For example, for GSM events, set **OutputPath** to


```
./data/backout/out/gsm/telephony.
```

Sample `OUT_GenericStream` registry for GSM events:

```
BackOutOnlyTELOutput
{
    ModuleName = OUT_GenericStream
    ProcessType = ALL_BCKOUT
    EventType = /event/delayed/session/telco/gsm
    Module
    {
        Grammar = ./formatDesc/Formats/Solution42/V670_EVENT_LOADER_
OutGrammar.dsc
        DeleteEmptyStream = True
        OutputStream
        {
            ModuleName = EXT_OutFileManager
            Module
            {
                OutputPath = ./data/backout/out/gsm/telephony
                OutputPrefix = test_TEL
                OutputSuffix = .out
                TempPrefix = .
                TempDataPath = ./data/backout/out/gsm/telephony
                TempDataPrefix = tel.tmp.
```

```

        TempDataSuffix = .data
        Replace = TRUE
    }
}
}
}

```

Sample OUT_GenericStream registry for GPRS event:

```

BackOutOnlyGPRSOutput
{
    ModuleName = OUT_GenericStream
    ProcessType = ALL_BCKOUT
    EventType = /event/delayed/session/gprs
    Module
    {
        Grammar = ./formatDesc/Formats/Solution42/V670_EVENT_LOADER_
OutGrammar.dsc
        DeleteEmptyStream = True #defaults to true
        OutputStream
        {
            ModuleName = EXT_OutFileManager
            Module
            {
                OutputPath = ./samples/wireless/data/backout/out/gprs
                OutputPrefix = test_GPRS
                OutputSuffix = .out
                TempPrefix = .
                TempDataPath = ./samples/wireless/data/backout/out/gprs
                TempDataPrefix = gprs.tmp.
                TempDataSuffix = .data
                Replace = TRUE
            }
        }
    }
} # end of BackOutOnlyGPRSOutput

```

Important: To ensure output file integrity, specify a unique combination of OutputPath, OutputSuffix, and OutputPrefix values for each output stream defined in the registry.

About Synchronizing Rating and Loading Applications

When you use Pipeline Manager to rerate wireless events and load those events using RE Loader, you must synchronize the following applications:

- Pipeline Manager
- Event Extraction
- Rated Event Loader

You must synchronize these applications for two reasons:

- RE Loader locks the BRM database tables that it loads into, making them inaccessible to other applications during the loading process. Event Extraction Manager returns an error if it cannot access the database tables.

- If any events have not been rated and loaded before Pipeline Manager starts rerating, the account balances can be incorrectly updated when those events are processed after rerating is complete.

You must synchronize these applications manually by stopping and restarting them in a specific order:

1. Ensure that all events have been rated by Pipeline Manager.
2. Ensure that all rated events have been loaded by RE Loader.
There should be no remaining unrated files in the pipeline output directories. If errors occurred during loading that require you to reload events, perform those tasks before continuing.
3. Stop the rerating pipeline.
4. Stop the Batch Controller, thereby disabling any scheduled RE Loader processes.
5. Run the Event Extraction utility.
6. Start the backout pipeline to remove all the erroneous impact balances from the discount accounts.
7. Stop the backout pipeline and restart the rerating pipeline to rerate the events. The rerating pipeline can be restarted using the Pipeline Manager EventHandler.
8. If you configured RE Loader to run *manually*, run the RE Loader `"pin_rel"` utility. If errors occur during loading, correct them and reload the events before continuing.
9. If you configured RE Loader to run *automatically*, enable the REL handler that runs `pin_rel` in the Batch Controller configuration file by doing the following:
 - Stop and restart the Batch Controller to run `"pin_rel"` and load the rerated events. If errors occur during loading, correct them and reload the events before continuing.
 - Disable the REL handler that runs `pin_rerel` in the Batch Controller configuration file.
10. Enable RE Loader in the Batch Controller configuration file.

Using Event Extraction Manager

This chapter describes how to use Oracle Communications Billing and Revenue Management (BRM) Event Extraction Manager to extract prerated events for rerating.

Important: Event Extraction Manager is packaged with Rated Event (RE) Loader.

Before extracting events, you must know the following:

- Basic BRM concepts
- BRM system architecture
- How to create and edit BRM configuration files

About Event Extraction Manager

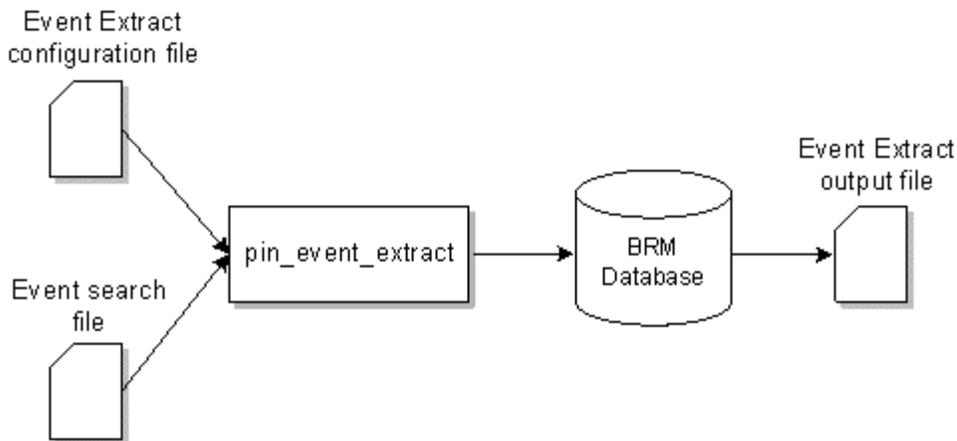
Event Extraction Manager is the first application used in the rerating process. You use it to find and extract events from the BRM database that must be rerated by Pipeline Manager and then reloaded into the BRM database by Rated Event (RE) Loader. For an overview, see "[About Rerating Pipeline-Rated Events](#)".

Event Extraction Manager does not modify or process events, nor does it lock accounts in your database. This enables BRM and other applications to safely access accounts in the database while you are extracting events.

Event Extraction Manager consists of the following components:

- The event extract configuration file (**pin.conf**), which specifies how the **pin_event_extract** utility connects to your BRM system and the size and name of the event extract output file
- The **pin_event_extract.cfg** file, which specifies the event search criteria
- The **pin_event_extract** utility, which searches the BRM database for events meeting the search criteria and writes the results to an output file
- The event extract output file, which contains the list of events to rerate

[Figure 26–1](#) shows Event Extraction Manager:

Figure 26–1 Event Extraction Manager

About the Event Extract Configuration File

The event extract configuration file (**pin.conf**), which specifies how the **pin_event_extract** utility connects to your BRM system, and the size and name of your event extract output file.

If you are extracting events from a multischema system, you must modify this file for each database schema in your system. For information, see ["Extracting Events from a Multischema System"](#).

For information on how to create this file, see ["Configuring Connection and Output File Parameters"](#).

About the pin_event_extract.cfg File

The event search file (*BRM_home/apps/pin_event_extract/pin_event_extract.cfg*) specifies the criteria that the **pin_event_extract** utility uses to search for events in your BRM database. You can search for events based on a wide variety of criteria, including account number, product name, and impact category.

For information about the event search criteria, see ["Event Search Criteria"](#).

For information on creating a **pin_event_extract.cfg** file, see ["Specifying Which Events to Extract for Rerating"](#).

About the pin_event_extract Utility

The **pin_event_extract** utility builds a search query with the criteria specified in the **pin_event_extract.cfg** file and then run it against the specified database schema. The search query selects only events that have a balance impact type of pipeline-processed, indicating that Pipeline Manager has already rated the events.

About the Event Extract Output File

The **pin_event_extract** utility prints to a file the list of events meeting the search criteria. Results are written in tab-delimited columns conforming to the Oracle CDR format. You send this file directly to Pipeline Manager for rerating.

The default implementation defines the event-field-to-output file mapping for GSM and GPRS events. However, you can create additional categories by modifying the **PCM_OP_BILL_POL_CONFIG_EET** policy opcode.

When the number of events meeting the search criteria exceeds the specified maximum file size or number of EDRs, the utility generates multiple output files and appends a number, such as `_1`, `_2`, and so on, to each file name.

When you use the sequence number search criteria, the utility automatically appends a sequence number to the file name and ignores the specified maximum file size and number of EDRs.

Event Search Criteria

Event Extraction Manager searches for events by using the criteria in [Table 26–1](#).

To specify which events to extract:

- To extract events generated for a specific brand, run the `pin_event_extract` utility with the `-b` parameter.
- To extract events using all other criteria, edit the `pin_event_extract.cfg` file.

Important: The event start time, event end time, and event type criteria are required.

Table 26–1 Event Search Criteria

Criteria	Entry Name	Description	Required
account number	account	<p>Extracts events generated by the specified account.</p> <p>Caution: If you use this criteria, ensure the following:</p> <ul style="list-style-type: none"> ■ All events for the specified accounts are available for rerating ■ All specified accounts reside in the same database schema ■ The event start time, event end time, and event type criteria are required. 	no
brand name	brand	<p>Extracts events generated for the specified brand. If you do not specify a brand, the utility uses the default root account.</p> <p>Note: To extract events by brand, you must run the <code>pin_event_extract</code> utility with the <code>-b</code> parameter.</p>	no

Table 26–1 (Cont.) Event Search Criteria

Criteria	Entry Name	Description	Required
deal name	deal	Extracts events generated by all products and discounts associated with the specified deal. Caution: Do not use this criteria with the product or rate plan search criteria. Doing so prevents the utility from extracting all events associated with the specified deal.	no
event occurrence start and end times	event_start_time_stamp event_end_time_stamp	Extracts events that occurred after the specified starting time stamp and before the specified ending time stamp. The start and end times are inclusive.	yes
event creation start and end times	event_created_start_time_stamp event_created_end_time_stamp	Extracts events loaded into the database after the specified creation start time stamp and before the specified creation end time stamp. The start and end times are inclusive.	no
event type	event_category	Extracts the specified event type. By default, this criteria is set to extract the /event/delayed/session/elco/gsm event type.	yes
G/L ID number	gl_id	Extracts events with the specified general ledger G/L ID.	no
impact category name	impact_category	Extracts events rated with the specified impact category.	no
item object number	item_obj	Extracts events generated for the specified item.	no
product name	product	Extracts events generated for the specified products. Caution: Do not use this criteria with the deal or rate plan search criteria.	no
rate plan name	rate_plan	Extracts events rated by the specified rate plan. Caution: Do not use this criteria with the deal or product search criteria.	no

Table 26-1 (Cont.) Event Search Criteria

Criteria	Entry Name	Description	Required
resource ID	resouce_id	Extracts events with the specified resource ID. This criteria works best when the resource ID represents a noncurrency asset such as Internet hours.	no
search event field	search_event_type search_field search_parameter	<p>Extracts events of type <i>search_event_type</i> where <i>search_field</i> contains <i>search_parameter</i>. For example, you can use this search criteria to extract events based on custom fields.</p> <ul style="list-style-type: none"> ▪ <i>search_event_type</i> specifies the type of event to search. Only events of this type or a subclass of this event type are extracted <p>Important: All events of type <i>search_event_type</i> must contain the <i>search_field</i>. You must ensure that the storable class specified by the event type contains <i>search_field</i>.</p> <ul style="list-style-type: none"> ▪ <i>search_field</i> specifies the field in the event used for event extraction. Only those events where <i>search_field</i> is equal to <i>search_parameter</i> are extracted. ▪ <i>search_parameter</i> specifies the value of <i>search_field</i>. <p>Note: <i>search_parameter</i> is used in addition to other search criteria specified in the pin_event_extract.cfg file.</p>	no

Table 26–1 (Cont.) Event Search Criteria

Criteria	Entry Name	Description	Required
sequence number	file_sequence_number	<p>Extracts events processed with the specified file sequence number. This number is assigned by Pipeline Manager and then added to the event object when RE Loader loads the event into the database. This enables you to extract and rerate events that were processed at a particular time.</p> <p>You can find an event's file sequence number in the <code>/batch/rel</code> object.</p> <p>When you use this criteria, the <code>pin_event_extract</code> utility automatically appends the file sequence number to the output file name.</p> <p>Note: You can specify any sequence number other than 0. RE Loader uses 0 to indicate that Pipeline Manager did not assign a sequence number to the event.</p>	no
service name	service	<p>Extracts events generated for the specified service.</p> <p>Important: The specified service must correspond to a unique service type.</p>	no

Synchronizing Extraction and Rating Applications

It is important to rate all events before running Event Extraction Manager. If there are events not yet rated when you run Event Extraction Manager, your account balances will not be correct.

Event Extraction Manager and RE Loader use the same status table, `REL_EVENT_EXTRACT_SYNC_T`, to check if one of the other applications is running. If you attempt to start Event Extraction Manager while RE Loader is running, Event Extraction Manager fails to start and returns an error message.

However, if Event Extraction Manager or RE Loader terminate abnormally, the status table may be incorrectly left in a locked state. You must use the `pin_event_extract` utility's override option before you can extract events.

For more information, see "[About Synchronizing Rating and Loading Applications](#)".

Extracting Events

Extracting events includes the following tasks:

1. [Configuring Connection and Output File Parameters](#)
2. [Specifying Which Events to Extract for Rerating](#)
3. [Running the pin_event_extract Utility](#)
4. [Troubleshooting Event Extraction Errors](#)
5. [Sending the Output File to Pipeline Manager](#)

Configuring Connection and Output File Parameters

To configure your connection and output file parameters:

1. Open the event extract configuration file (*BRM_home/apps/pin_event_extract/pin.conf*) in a text editor.
2. Edit the connection and log file entries.
3. Specify the EDR output file name:

```
- pin_event_extract filename SOL42_SenderReceiver
```

where *Sender* is the code for the sender of the file and *Receiver* is the code for the recipient of the file. For example, if the sender's code is D00D1 and the receiver's code is SOL42, replace *SenderReceiver* with **D00D1SOL42**.

Note:

- When you use the sequence number search criteria, the utility automatically appends the file sequence number to the file name.
 - When the output exceeds the specified file maximums, the utility automatically generates multiple output files and appends a number to each file name.
-
-

4. Specify the UTC Time Offset: where UTC Time Offset is the difference between the local time and UTC time:

```
- pin_event_extract UTCoffset Value
```

5. Specify the specification version number:

```
- pin_event_extract specvernum VersionNumber
```

6. Specify the specification release version number:

```
- pin_event_extract specrelver ReleaseNumber
```

7. Specify the maximum number of EDR creation threads:

```
- pin_event_extract num_threads MaximumNumber
```

8. Specify the number of EDR records Event Extraction Manager retrieves during each step search:

```
- pin_event_extract step_size SearchNumber
```

9. Specify the maximum size, in bytes, of the event extract output file:

```
- pin_event_extract maxfilesize FileSize
```

When the file size exceeds the specified maximum, the `pin_event_extract` utility generates multiple output files and appends a number, such as `_1`, `_2`, and so on, to each file name.

10. Specify the maximum number of EDRs allowed in each output file.

```
- pin_event_extract maxEDRs MaximumEDRs
```

When the number of EDRs exceeds the specified maximum, the `pin_event_extract` utility generates multiple output files and appends a number, such as `_1`, `_2`, and so on, to each file name.

11. Save and close the file.

Specifying Which Events to Extract for Rerating

The `pin_event_extract` utility finds the events for rerating by creating a search query. You specify the parameters for the search in the `BRM_home/apps/pin_event_extract/pin_event_extract.cfg` file.

To extract events:

1. Open the `BRM_home/apps/pin_event_extract/pin_event_extract.cfg` file in a text editor.
2. Set the `event_start_time_stamp` and `event_end_time_stamp` entries.

```
event_start_time_stamp MM/DD/YYYY [ hh:mm:ss ]
event_end_time_stamp MM/DD/YYYY [ hh:mm:ss ]
```

where `hh:mm:ss` time in 24-hour mode. For example, enter `16:00:00` to specify 4:00 p.m. If you do not specify `hh:mm:ss`, the time defaults to midnight (00:00:00).

The `pin_event_extract` utility extracts events that occurred between the specified start date and time and the end date and time (inclusive).

3. Set the `event_category` entry:

```
event_category category
category EventTypeName
```

where:

- `category` specifies the type of event to extract. The default implementation includes the `gsm` and `gprs` event categories only, but you can create additional categories by modifying the `PCM_OP_SUBSCRIPTION_POL_CONFIG_EET` policy opcode.
- `EventTypeName` specifies the name of the event you want extracted. For example, `/event/delayed/activity/gprs`.

Tip: You can list multiple event types on one line by using colons (`:`) or semicolons (`;`) to delimit events.

4. Set your search criteria. The `pin_event_extract.cfg` file includes examples and instructions.

For an overview, see "[Event Search Criteria](#)".

5. Save and close the file.

Extracting Events from Specific Partitions

Events are loaded into database tables that are partitioned by a period such as days, weeks, or months. You can improve event extraction performance by limiting the number of partitions from which events are extracted. To do this, specify the start and end dates of when the events were loaded into the database.

For example, if your database tables are partitioned by months, you can extract events from a single partition by specifying the start and end day of a single month.

To extract events from specific partitions:

1. Open the `BRM_home/apps/pin_event_extract/pin_event_extract.cfg` file in a text editor.
2. Set the `event_created_start_time_stamp` and `event_created_end_time_stamp` entries. The `pin_event_extract` utility extracts events that were loaded into the database on or after the start time and on or before the end time.

```
event_created_start_time_stamp MM/DD/YYYY hh:mm:ss
event_created_end_time_stamp MM/DD/YYYY hh:mm:ss
```

where:

- `MM/DD/YYYY` is the month, day, and year.
- `hh:mm:ss` is the time in 24-hour mode. For example, enter `16:00:00` to specify 4:00 p.m.

Note: Entering only a day returns an error.

Sample `pin_event_extract.cfg` File for a Specific Rate Plan

The following sample `pin_event_extract.cfg` file extracts events that meet the following criteria:

- Created between 6:00 p.m. on June 1, 2002, and 6:00 p.m. on August 30, 2002.
- Is the following prerated GSM event: `/event/delayed/session/telco/gsm`.
- Rated with the Free Saturdays rate plan.

```
event_start_time_stamp 06/01/2002 18:00:00
event_end_time_stamp 08/30/2002 18:00:00

event_category gsm
gsm /event/delayed/session/telco/gsm

rate_plan Free Saturdays
```

Sample `pin_event_extract.cfg` File for a Specific Sequence Number

The following sample `pin_event_extract.cfg` file extracts events that meet the following criteria:

- Created between 10:00 a.m. on January 1, 2002 and 9:59:59 a.m. on January 31, 2002.
- Represents a prerated GPRS event
- Processed with the 777888999 sequence number

```
event_start_time_stamp 01/01/2002 10:00:00
event_end_time_stamp 01/31/2002 09:59:59
```

```
event_type /event/delayed/session/gprs  
file_sequence_number 777888999
```

Running the `pin_event_extract` Utility

The `pin_event_extract` utility generates a search query with your specified search criteria.

To extract events:

1. Ensure that all events that you want to extract have been processed.
2. Ensure that RE Loader is not running.

For information, see "[Synchronizing Extraction and Rating Applications](#)".

Note: If RE Loader and `pin_event_extract` are run simultaneously, `pin_event_extract` extracts only those records that are completely processed by RE Loader.

3. Run the following command:

```
pin_event_extract [-f ConfigFileName] [-b]
```

Troubleshooting Event Extraction Errors

Event extraction may fail for several reasons:

- RE Loader is currently running, preventing Event Extraction Manager from starting. Wait for RE Loader to finish loading events before restarting Event Extraction Manager.
- The status table was incorrectly locked, preventing Event Extraction Manager from starting. This can occur when one of the applications terminates abnormally and cannot reset the status table before exiting. To reset the status table:

Caution: Do not use this option unless you are certain that RE Loader is stopped.

```
pin_event_extract -o TRUE
```

You can check the event extract log file, created by default in the `BRM_home/var/pin_event_extract` directory, for information on why extraction failed.

After you fix the problem, you can safely rerun the `pin_event_extract` utility with the same `pin_event_extract.cfg` file.

Sending the Output File to Pipeline Manager

To begin rerating events, send the event extract output file to Pipeline Manager for processing. For information, see "[About Rerating Pipeline-Rated Events](#)".

Extracting Events from a Multischema System

To extract events from a multischema system:

1. On the system on which Event Extraction Manager is installed, open the *BRM_home/apps/pin_event_extract/pin.conf* in a text editor.
2. Specify which events to extract by using the *pin_event_extract.cfg* file.
3. Run the *pin_event_extract* utility and fix any errors.
4. Change the connection and output file parameters for the next schema.
5. Run the *pin_event_extract* utility and fix any errors.
6. Repeat steps 4 and 5 until you have extracted events from all schemas.

Customizing How to Extract Events for Rerating

You can use the `PCM_OP_SUBSCRIPTION_POL_CONFIG_EET` policy opcode to customize which event fields are extracted for rerating; for example, you can modify which GPRS fields are extracted, or you can define the mapping for a custom event category.

The `PCM_OP_SUBSCRIPTION_POL_CONFIG_EET` policy opcode defines, for each event category, the event fields that Event Extraction Manager passes to the event extract output file. The default implementation defines the event-field-to-output-file mapping for GSM and GPRS event categories only.

This policy opcode is called by Event Extraction Manager worker threads when the tool returns a list of POIDs that meet the search criteria.

By default, the `PCM_OP_SUBSCRIPTION_POL_CONFIG_EET` policy opcode takes an event's POID and category name as input and performs the following functions:

- Reads the entire event from the BRM database
- Maps predefined fields for GSM and GPRS events to a record buffer in the order received

Note: The *pin_event_extract* utility writes the contents of the record buffer to the event extract output file.

You can customize the `PCM_OP_SUBSCRIPTION_POL_CONFIG_EET` policy opcode to pass extended fields to the event extract output file. For example, you can modify which GPRS fields are passed to the output file, or you can define the mapping for a custom event category.

You can do the following:

- Read any additional event data based on the given POID
- Handle any new input formats: the policy opcode is written for the Oracle CDR format
- Define the event-field-to-output-file mapping for any new categories
- Write data to the correct position in the record buffer

Configuring Rerating in Pipeline Manager

This chapter describes how to rerate events by using the Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager.

For information about rerating, see ["About Rerating Pipeline-Rated Events"](#).

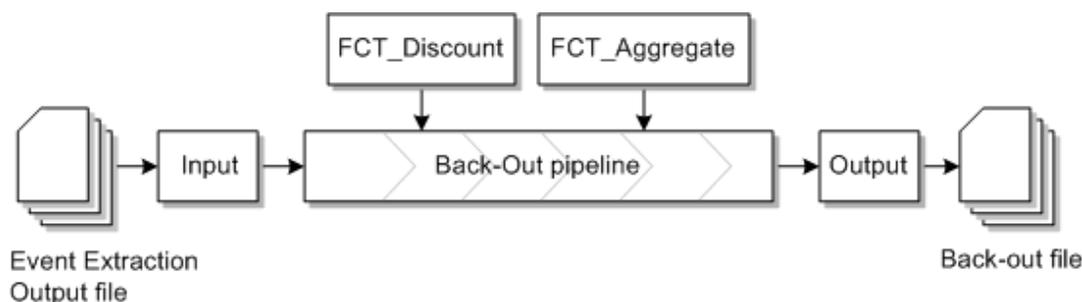
About the Back-Out Pipeline

The back-out pipeline is a separate pipeline and contains the following modules:

- The FCT_Discount module removes the EDR discount balance impacts from the discount accounts.
- The FCT_Aggregate module calculates the aggregation results and writes them to the back-out result file. It writes one result file for each transaction.

Figure 27-1 shows the back-out pipeline:

Figure 27-1 Back-Out Pipeline



Configuring the Back-Out Pipeline

When you configure the back-out pipeline, you configure the following parameters in the pipeline registry:

- The ISC_AddCBD iScript. You configure this by configuring an instance of the FCT_IScript module.
- The FCT_Discount module. You set the **BackOut** registry entry to TRUE.
- The FCT_Aggregate module. You set the **BackOut** registry entry to TRUE.

Pipeline Manager creates one control file and one result file for each transaction and each aggregation scenario.

About Starting the Back-Out Pipeline

Before you start the back-out pipeline stop all rating pipelines and ensure that there are no outstanding events being loaded. See "[About Synchronizing Rating and Loading Applications](#)".

You start the back-out pipeline by sending a semaphore:

```
ifw.Pipelines.ALL_BCKOUT.Active = TRUE
```

About Stopping the Back-Out Pipeline

You stop the back-out pipeline by sending a semaphore:

```
ifw.Pipelines.ALL_BCKOUT.Active = FALSE
```

About the Rerating Pipeline

The rerating pipeline contains the same function modules as the rating pipeline, except that it does not contain the FCT_CallAssembling and FCT_DuplicateCheck modules. The rerating pipeline locks the appropriate customer accounts until rerating is completed.

Note: Before you start the rerating pipeline, ensure that you corrected all configuration data in the Pipeline Manager database, such as the rate plan that caused the incorrect calculation.

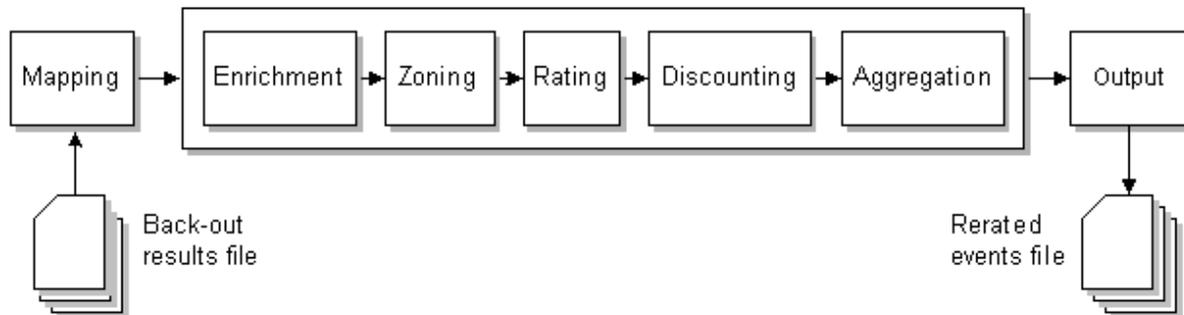
After the EDRs are re-rated with the updated configuration data, the FCT_BillingRecord module writes to the BRM billing record the difference between the old and re-rated balance impacts for each EDR.

The rerating output file contains all events for which the price has changed as a result of rerating and the change in the balance impact.

The output module sends the output file with the updated impact balances to the Rerated Event Loader (RREL) which loads them into the BRM database.

Figure 27–2 shows the rerating pipeline:

Figure 27–2 Rerating Pipeline



About Configuring the Rerating Pipeline

When you configure the rerating pipeline, you configure the same modules in the registry as for the normal rating pipeline. The only difference is that you do not configure the FCT_DuplicateCheck and the FCT_CallAssembling modules.

For the rerating pipeline, you use special backout input and output grammar files, and you create separate input and output directories.

For a sample registry for the rerating pipeline, see the **rerating.reg** file.

This sample shows the input configuration for a rerating pipeline. It uses a special backout grammar file, and a separate input directory.

```
Input
{
    UnitsPerTransaction = 1
    InputModule
    {
        ModuleName = INP_GenericStream
        Module
        {
            DefaultOutput = RerateOutput
            Grammar = ./formatDesc/Formats/Solution42/SOL42_V630_REL_InGrammar_
BACKOUT.dsc
            InputStream
            {
                ModuleName = EXT_InFileManager
                Module
                {
                    InputDirEmptyTimeout = 10
                    InputPath = ./samples/wireless/data/backout/in
                    InputPrefix = test.
                    InputSuffix = .edr
                    DonePath = ./samples/wireless/data/backout/done
                    DonePrefix = test.
                    DoneSuffix = .done
                    ErrorPath = ./samples/wireless/data/backout/err
                    ErrorPrefix = test.
                    ErrorSuffix = .err
                    TempPrefix = tmp.
                    Replace = TRUE
                }
            } # end of InputStream
        }
    } # end of InputModule
} # end of Input
```

This sample shows the output for telephony EDRs in a rerating pipeline. It uses a special backout output grammar file and a separate output directory.

```
TELOutput
{
    ModuleName = OUT_GenericStream
    Module
    {
        Grammar = ./formatDesc/Formats/Solution42/SOL42_V630_REL_OutGrammar.dsc
        DeleteEmptyStream = True
        OutputStream
        {
            ModuleName = EXT_OutFileManager
            Module
```

```
    {
      OutputPath = ./samples/wireless/data/rerate/telout
      OutputPrefix = test
      OutputSuffix = .out
      TempPrefix = .
      TempDataPath = ./samples/wireless/data/rerate/telout
      TempDataPrefix = tel.tmp.
      TempDataSuffix = .data
      Replace = TRUE
    }
  }
} # end of TELOutput
```

Important: To ensure output file integrity, specify a unique combination of OutputPath, OutputSuffix, and OutputPrefix values for each output stream defined in the registry.

About Starting the Rerating Pipeline

Before you start the rerating pipeline, ensure that the back-out pipeline has processed all EDRs that were extracted by the Event Extraction Tool. Also ensure that all other rating pipelines are stopped.

To start the rerating pipeline, use the event handler to start the rerating pipeline when the input directory is empty. To do so, configure the corresponding section in the registry. The EVT_INPUT_DIR_EMPTY event is triggered by the EXT_InFileManager when the input directory is emptied. In this sample, the event starts the rerating pipeline.

```
EventHandler
{
  ModuleName = EVT
  Module
  {
    Events
    ifw.Pipelines.ALL_BCKOUT.Input.InputModule.Module.InputStream.Module
    {
      EVT_INPUT_DIR_EMPTY = ./start_rerate.sh
    }
  }

  Buffer
  {
    Size = 100
  }
}
```

To define when to send EVT_INPUT_DIR_EMPTY event, use the EXT_InFileManager **InputDirEmptyTimeout** registry entry.

About Stopping the Rerating Pipeline

You stop the rerating pipeline manually by sending a semaphore:

```
ifw.Pipelines.RERATE.Active = FALSE
```

About Comprehensive Rerating Using `pin_rerate`

This chapter provides an overview of rerating using the Oracle Communications Billing and Revenue Management (BRM) "`pin_rerate`" utility to rerate real-time-rated events and pipeline-batch-rated events.

Note: This documentation does not apply to rerating only pipeline-batch-rated events by using a batch rerating pipeline.

For general information about rerating, see "[About Rerating Events](#)".

About Comprehensive Rerating

The `pin_rerate` utility provides a comprehensive rerating solution: You can rerate both real-time-rated and pipeline-batch-rated events concurrently or, if you do not use Pipeline Manager for batch rating, you can rerate only real-time-rated events. You can select accounts and events for rerating based on any event criteria. And you can configure BRM to automatically rerate certain accounts and events when you run `pin_rerate`.

The "`pin_rerate`" utility controls the rerating workflow. It follows a different rerating process depending on whether batch pipeline rating is enabled. See:

- [About Rerating Real-Time-Rated and Pipeline-Rated Events Concurrently](#)
- [About Rerating Events When You Do Not Use Pipeline Batch Rating](#)

About Rerate Jobs

A rerate job is a set of objects in the BRM database used to track the accounts selected for rerating and the status of the rerating process. Rerate jobs are created for all accounts to be rerated by `pin_rerate`. (For more information about rerate job objects, see "[How BRM Creates Rerate Jobs](#)".)

Rerate jobs are typically associated with a batch of accounts, but can also be associated with a single account.

Rerate jobs are created manually when you use `pin_rerate` to select accounts for rerating. Rerate jobs are created automatically when certain events occur, such as when backdated events or rate changes trigger automatic rerating.

When you use **pin_rerate** to select accounts for rerating, the utility assigns accounts to each rerate job and creates 2 rerate jobs per transaction. This enables **pin_rerate** to more easily roll back a transaction if an error occurs.

You can configure the number of accounts assigned to each rerate job and the number of jobs to create per transaction. See "[Configuring the Number of Accounts Per Job and Number of Jobs per Transaction](#)".

About Rerating Real-Time-Rated and Pipeline-Rated Events Concurrently

When you use **pin_rerate** to rerate both real-time-rated and pipeline-batch-rated events concurrently, the following tasks are performed:

1. The utility retrieves the accounts for rerating from the BRM database based on search criteria that you specify.
2. BRM extracts the events associated with those accounts and then backs out the original event balance impacts.
3. BRM determines whether the original event was rated in real time or by the batch rating pipeline:
 - a. It sends events previously rated in real time to the real-time rating opcodes for rating and discounting.
 - b. It sends events previously rated by a batch pipeline to the real-time rerating pipeline for rating and discounting. See "[About the Real-Time Rerating Pipeline](#)".
4. BRM records the rerated events in the BRM database.

About the Real-Time Rerating Pipeline

The **pin_rerate** utility sends events previously rated by the batch rating pipeline to a real-time rerating pipeline to be rerated.

The **pin_rerate** utility routes the pipeline-rated events to the Connection Manager (CM). The CM sends the pipeline-rated events in the form of a flist to the NET_EM pipeline module. The NET_EM module transfers the event to the real-time rerating pipeline for rerating.

The real-time rating opcodes add to the flist all the enrichment data needed by the real-time rerating pipeline to rate the event. For example, if the real-time rating opcodes determine that a discount should be applied, it adds the discount information to the flist.

Note: Real-time rerating process does not check credit limits when applying discounts. To configure rerating to check credit limits, create a custom iScript that sets the `DETAIL.CREDIT_LIMIT_CHECK` EDR field to `1`. See "Creating iScripts and iRules" in *BRM Developer's Guide*.

Similar to a batch rerating pipeline, a real-time rerating pipeline performs both rating and discounting functions.

To rerate events, the real-time rerating pipeline gets the new pricing information from the Pipeline Manager database. Certain configuration data, such as currency and non-currency resource information, are obtained from the BRM database.

The real-time rerating pipeline performs the following tasks:

1. The INP_Realttime input module converts the flist received from the NET_EM module to event data record (EDR) format and creates an EDR container.

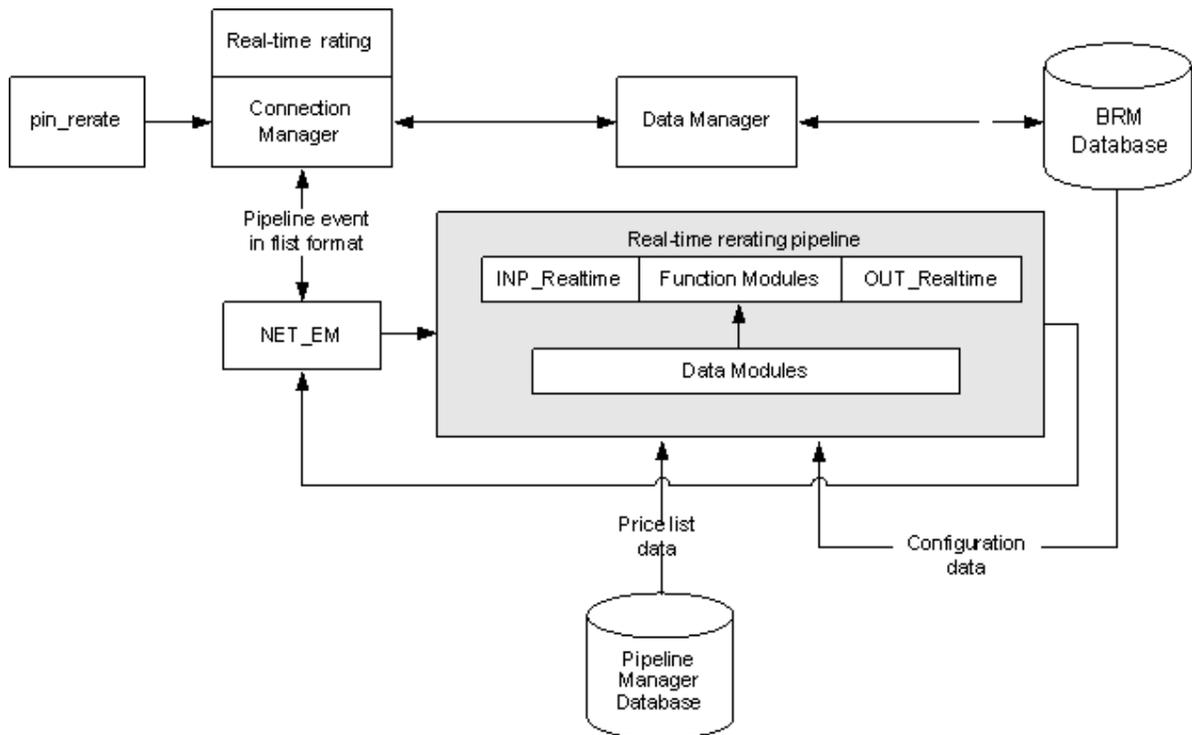
Tip: You can also create and run an iScript to manipulate data in the EDR container before it is sent to the pipeline modules. See "Creating iScripts and iRules" in *BRM Developer's Guide*.

2. The pipeline function modules calculate the new balance impacts by using the new pricing and discounting information and then adds the balance impacts to the EDR container.
3. The OUT_Realttime module converts the EDR back into flist format and sends the flist to NET_EM.
4. NET_EM sends the flist with the new balance impacts back to the CM.
5. BRM updates the account's balances and records the event in the BRM database.

For detailed information about configuring the real-time rerating pipeline, see "[Configuring a Real-Time Rerating Pipeline](#)".

Figure 28–1 shows the data flow for rerating events by using the real-time rerating pipeline:

Figure 28–1 Real-Time Rerating Pipeline Data Flow



About Transaction Management for the Real-Time Rerating Pipeline

The real-time rerating pipeline does not use the Transaction Manager (TAM) module. Instead, transaction handling is provided by the CM. When a rerating transaction fails, the CM rolls back the transaction and restores all the account balances in the BRM

database. For this reason, function modules and iScripts used by the real-time rerating pipeline are stateless.

Important: If you use iScripts for custom processing, ensure that the iScripts are stateless.

How `pin_rerate` and the Batch-Rating Pipeline Synchronize Processes

While pipeline-rated events are being rerated, the batch pipeline temporarily suspends new EDRs for the accounts that are currently being rerated. When rerating is complete, account balance data in Pipeline Manager is synchronized with the account balance data in the BRM database, and the batch pipeline resumes processing the suspended EDRs using the updated data.

The batch pipeline and `pin_rerate` use the Account Synchronization DM and BRM standard recycling to synchronize processes.

When you run `pin_rerate` for concurrent rerating, the following actions are taken to synchronize `pin_rerate` and batch pipeline processes:

1. The `pin_rerate` utility retrieves the accounts to be rerated and creates the rerate jobs. The rerate jobs are queued until they are processed.
2. `pin_rerate` sends a business event through the Account Synchronization DM to notify the batch pipeline that rerating for the selected accounts is about to begin. When it receives the notification, the batch pipeline suspends rating incoming CDRs for the accounts and sends `pin_rerate` a corresponding acknowledgment event.
3. `pin_rerate` rerates the accounts in the rerate job and BRM records the new balance impacts of the rerated events in the BRM database.
4. `pin_rerate` sends a business event through the Account Synchronization DM to notify the batch pipeline that rerating is complete for the selected accounts. The batch pipeline synchronizes the account data in pipeline memory with the account data in the BRM database and resumes processing EDRs. The batch pipeline sends `pin_rerate` a corresponding acknowledgment event.
5. `pin_rerate` recycles the EDRs suspended during the rerating process by calling the standard recycling opcode.

For more information about the Account Synchronization DM, see "About Account Synchronization" in *BRM Installation Guide*.

After successful completion of each step, `pin_rerate` updates the rerate job status so that the next step in the rerate workflow can be executed.

Rerate jobs that are processed during concurrent rerating can have the statuses shown in [Table 28-1](#):

Table 28-1 Status for Rerate Jobs

Status	Description
NEW	The initial status of a rerate job.
WAITING_ACCOUNT_LOCKED	The status after <code>pin_rerate</code> has notified the batch pipeline to suspend EDRs for accounts in the rerate job.
ACCOUNT_LOCKED	The status after <code>pin_rerate</code> has received acknowledgment from the batch pipeline.

Table 28–1 (Cont.) Status for Rerate Jobs

Status	Description
RERATED	The status after the events associated with the accounts in the rerate jobs have been rerated.
READY_FOR_RECYCLE	The status after account data has been synchronized between the BRM database and Pipeline Manager and when the batch pipeline has resumed processing EDRs for the accounts in the rerate job.
COMPLETE	The status when rerating is completed for <i>all</i> the accounts in the rerate job.

About Suspending and Recycling EDRs during the Rerating Process

To suspend EDRs for Accounts that are being rerated, the pipeline DAT_AccountBatch module sets a rerate error code and a recycle key value in the EDR. The error code causes the EDR to be suspended by BRM standard recycling. The recycle key is used to select the EDRs to be recycled.

For information about recycling, see "About the EDR Recycling Features" in *BRM Configuring Pipeline Rating and Discounting*.

You must load suspended EDRs into the BRM database by running Suspended Event (SE) Loader. You typically schedule SE Loader to run automatically when you set up standard recycling. For more information, see "About Suspended Event (SE) and Suspended Batch (SB) Loader" in *BRM Configuring Pipeline Rating and Discounting*.

Suspended EDRs are stored in the database until they are recycled. To recycle the EDRs that were suspended during the rerating process, you run **pin_rerate** with the **-process recycle** parameter after rerating is completed. For more information, see "Recycling EDRs Suspended during Rerating".

Procedure for Concurrent Rerating of Real-Time-Rated and Pipeline-Rated Events

To rerate both real-time-rated events and pipeline-batch-rated events concurrently, you perform the following steps:

1. Run **pin_rerate** to select the accounts for rerating and create rerate jobs. See "Selecting Accounts and Events for Rerating".

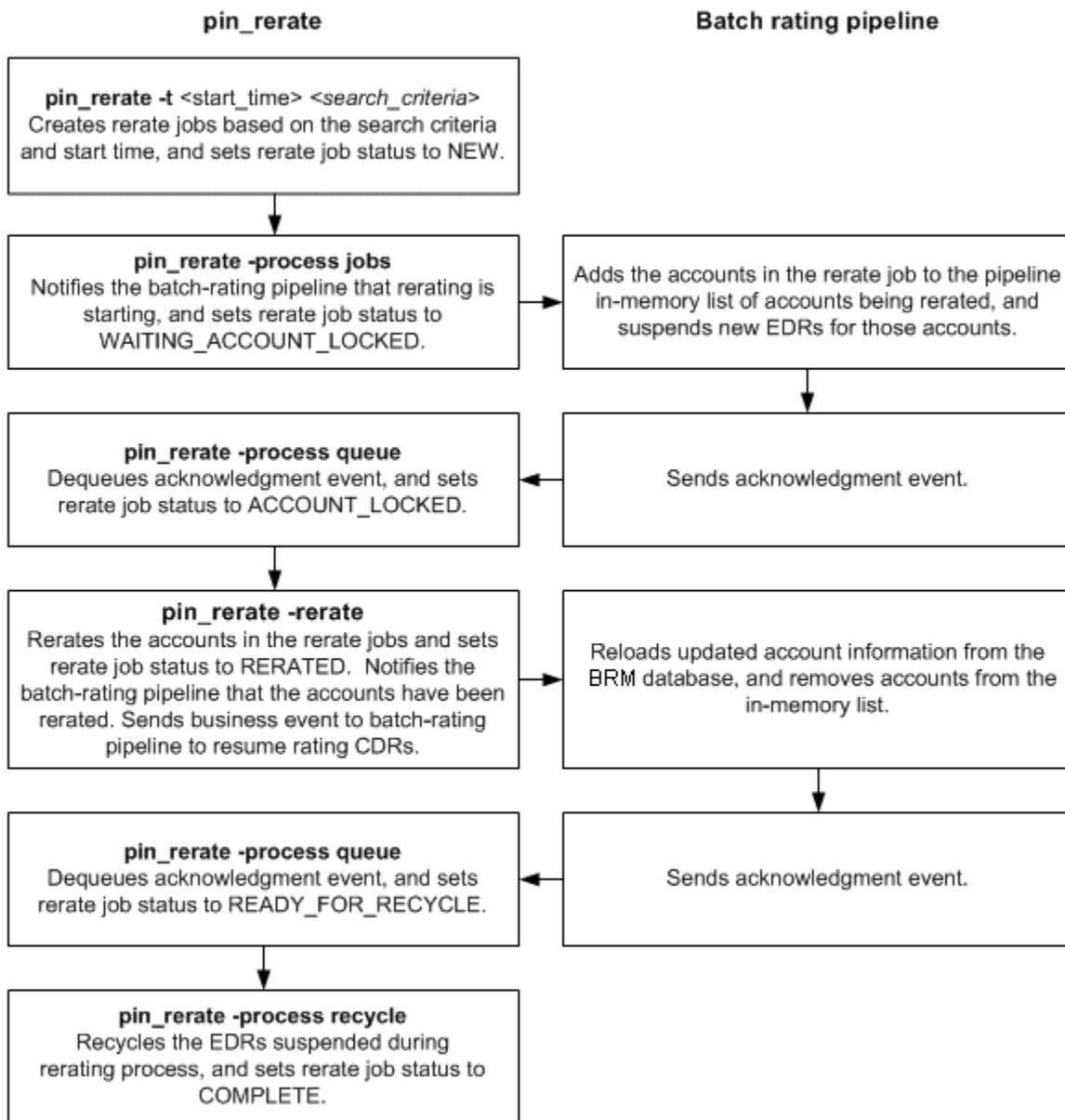
Note: Rerate jobs can also be created automatically through BRM automatic rerating or trigger-dependent rerating.

2. Run a series of **pin_rerate** commands to rerate the events associated with the accounts in the rerate job. See "Rerating Real-Time-Rated and Pipeline-Rated Events Concurrently".
3. Run **pin_rerate** to recycle the EDRs associated with the accounts in the rerate job that were suspended during rerating. See "Recycling EDRs Suspended during Rerating".

For more information, see "Using the pin_rerate Utility".

Figure 28–2 shows the rerating commands, process flow, and job status transitions when you perform concurrent rerating:

Figure 28–2 Concurrent Rerating Process Flow



About Rerating Events When You Do Not Use Pipeline Batch Rating

If you do not use Pipeline Manager for batch rating, you can use **pin_rerate** to rerate only real-time-rated events.

Note: Real-time-rated events are events that are rated by the BRM rating opcodes. Real-time-rated events include events that are loaded in batches by Universal Event (UE) Loader.

To rerate only real-time-rated events, you must configure the **batch_rating_pipeline** business parameter so that **pin_rerate** does not attempt to communicate with a batch

pipeline, which will cause rerating to fail. For more information, see ["Specifying Whether the Batch Rating Pipeline Is Enabled"](#).

When rerating only real-time-rated events, the following actions are taken when you run `pin_rerate`:

1. The `pin_rerate` utility selects the accounts to be rerated from the BRM database based on the specified search criteria and creates rerate jobs.

Note: Rerate jobs can also be created automatically through BRM automatic rerating or trigger-dependent rerating.

2. `pin_rerate` invokes rerating of the selected accounts in the rerate jobs.
3. BRM extracts the events associated with the selected accounts from the BRM database and does the following:
 - a. Rerates the real-time-rated events.
 - b. Records the new balance impacts of the rerated events in the BRM database.

After the successful completion of each step, `pin_rerate` updates the rerate job status so that the next step in the rerate workflow can be executed.

Rerate jobs that are processed when rerating only real-time-rated events can have the statuses shown in [Table 28–2](#):

Table 28–2 Status for Real-Time Rerate Events

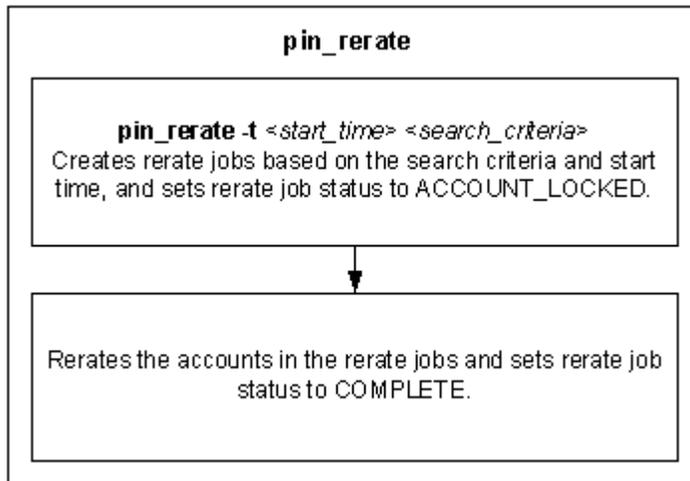
Status	Description
ACCOUNT_LOCKED	The initial status of rerate jobs.
COMPLETE	The status when rerating has completed for <i>all</i> the accounts in the rerate job.

Procedure for Rerating Only Real-Time-Rated Events

To rerate only real-time-rated events when you do not use Pipeline Manager for batch rating, you select the accounts to rerate and rerate the selected accounts using a single `pin_rerate` command. See ["Processing Rerate Jobs When You Do Not Use Pipeline Batch Rating"](#).

For more information about using `pin_rerate`, see ["Using the pin_rerate Utility"](#).

[Figure 28–3](#) shows the rerating command, process flow, and job status transitions when rerating only real-time-rated events:

Figure 28–3 Rerating Only Real-Time Events

How Failed Rerate Jobs Are Processed

The accounts in a rerate job are typically processed individually in separate rerate operations. When rerating fails for one or more accounts in a rerate job, **pin_rerate** sets the status of the accounts in the rerate job batch to FAILED. When rerating process is complete for the rerate job, **pin_rerate** creates a new rerate job consisting of only the accounts that failed. The new rerate job is processed the next time **pin_rerate** is run.

If an account selected for rerating is associated with a subscription service that was transferred during the period for which rerating is specified, then the account to which the service was transferred is included in the rerate job and all those accounts are rerated concurrently in a single rerate operation. When rerating fails for one of these accounts, then rerating fails for all accounts in the rerate request. In this case, **pin_rerate** creates a new rerate job containing all the accounts in the rerate request.

For example, subscription service X is originally owned by Account A and transferred to Account B on June 15. Later in the month, it is transferred from Account B to Account C. Rerating of Account A from June 1 also results in rerating of accounts B and C. Accounts A, B, and C are grouped together in a single rerate request. If rerating fails for any of these accounts, **pin_rerate** creates a new rerate job consisting of all three accounts in a single rerate request. The accounts are rerated again the next time **pin_rerate** is run.

For more information about rerating accounts associated with subscription service transfer, see ["Configuring Rerating for Accounts Associated With Subscription Service Transfer"](#).

About Automatic Rerating

When certain events occur in the BRM system (such as backdated purchase events), customer accounts require rerating so they are billed accurately. In automatic rerating, BRM automatically creates rerate jobs for these events.

Note: Automatic rerating *does not* immediately rerate accounts when the rerate job is created. You must still run the **pin_rerate** utility.

Accounts and events associated with rerate jobs that are created automatically are rerated the next time you run "[pin_rerate](#)" to process rerate jobs: you do not need to first select the accounts and events by specifying `pin_rerate` selection criteria. For more information, see "[About Processing Rerate Jobs Created by Automatic Rerating](#)".

You enable automatic rerating by configuring event notification to call the `PCM_OP_SUBSCRIPTION_POL_GENERATE_RERATE_REQUEST` policy opcode when events occur that require rerating.

Automatic rerating uses rerate reason codes. A rerate reason code is passed in to `PCM_OP_SUBSCRIPTION_POL_GENERATE_RERATE_REQUEST`, which uses the code to help determine whether rerating is required for an event.

BRM creates rerate jobs for automatic rerating as follows:

1. An event occurs that requires accounts to be rerated.
2. The opcode that triggers rerating when that particular event occurs creates a notification event.

[Table 28-3](#) summarizes the types of notification events used to trigger automatic rerating and the rerating scenario to which each event applies:

Table 28-3 *Types of Notifications for Automatic Rerating*

Notification Events for Automatic Rerating	Rerating Scenario
<code>/event/notification/auto_rerate</code>	Automatic rerating of backdated events. See " About Automatic Rerating of Backdated Events ".
<code>/event/notification/rate_change</code>	Automatic rerating for rate changes.
<code>/event/notification/rollover_correction/rerate</code>	Automatic rerating of rollover corrections due to delayed events.
An event by which you want rerating to be triggered	Automatic rerating for your custom rerating requirements. See " About Trigger-Dependent Rerating ".

3. The event notification mechanism calls `PCM_OP_SUBSCRIPTION_POL_GENERATE_RERATE_REQUEST`.
4. `PCM_OP_SUBSCRIPTION_POL_GENERATE_RERATE_REQUEST` takes as input the event type and the rerate reason code associated with the event and analyzes the event to determine if rerating is required.

Note: BRM reserves the range of 100 to 120 for automatic rerating reason codes.

5. If rerating is required, `PCM_OP_SUBSCRIPTION_POL_GENERATE_RERATE_REQUEST` calls the `PCM_OP_RERATE_INSERT_RERATE_REQUEST` opcode along with the appropriate rerating criteria to create a rerate job.

`PCM_OP_SUBSCRIPTION_POL_GENERATE_RERATE_REQUEST` calls `PCM_OP_SUBSCRIPTION_INSERT_RERATE_REQUEST` only when the reason code passed in is one of those reserved for automatic rerating (reason codes between 100 and 120). By default, if a different reason code is passed in, the opcode does nothing. To create rerate jobs for reason codes other than those reserved for automatic rerating, you must customize `PCM_OP_SUBSCRIPTION_POL_GENERATE_RERATE_REQUEST`.

By default, PCM_OP_SUBSCRIPTION_POL_GENERATE_RERATE_REQUEST passes a reason code of 0 (no rerate reason) to PCM_OP_SUBSCRIPTION_INSERT_RERATE_REQUEST. To rerate accounts separately based on a rerate reason code, customize PCM_OP_SUBSCRIPTION_POL_GENERATE_RERATE_REQUEST to pass the rerate reason code it receives (or a different rerate reason code). The rerate reason code is stored in the rerate job object, and you can select rerate jobs based on the specific rerate reason code when you run **pin_rerate** to process rerate jobs.

Note: You can also assign reason codes when you manually create rerate jobs by using **pin_rerate**. For more information, see ["Assigning Rerate Reasons to Rerate Jobs"](#).

For information about rerating accounts based on a reason code, see ["Rerating Real-Time-Rated and Pipeline-Rated Events Concurrently"](#) and ["Processing Rerate Jobs When You Do Not Use Pipeline Batch Rating"](#).

You can also customize PCM_OP_SUBSCRIPTION_POL_GENERATE_RERATE_REQUEST in other ways. For example, you can change whether rerate jobs are created for specific events: such as when you want to rerate events for only a few (rather than all) automatic rerating scenarios.

How Rerating Affects Account Migration

When an account is selected for rerating, the account cannot be migrated to another database until rerating is complete. Otherwise, the account is not rerated. For this reason, the Account Migration Manager (AMM) does not allow migration of an account to another database if the account is being rerated by **pin_rerate**.

The AMM does not migrate an account if the account is in a rerate job and the rerate job status is one of the following:

- WAITING_ACCOUNT_LOCKED
- ACCOUNT_LOCKED
- RERATED
- READY_FOR_RECYCLE

However, the account is migrated if the rerate job status is NEW. In this case, the AMM deletes the account from the rerate job in the source database and creates a new rerate job with the account in the destination database.

Note: After the account migration is complete, you must run **pin_rerate** in the destination database to rerate the account.

Managing Comprehensive Rerating with Custom Applications

BRM uses the following opcodes for comprehensive rerating with **pin_rerate**:

- PCM_OP_SUBSCRIPTION_RERATE_REBILL. See ["How Comprehensive Rerating Works"](#).
- PCM_OP_RERATE_INSERT_RERATE_REQUEST. See ["How BRM Creates Rerate Jobs"](#).
- PCM_OP_SUBSCRIPTION_PREP_RATE_CHANGE. See ["Tracking Rate Changes for Rerating"](#).

- PCM_OP_SUBSCRIPTION_RATE_CHANGE. See ["Rerating Cycle Fees"](#).
- PCM_OP_SUBSCRIPTION_POL_SPEC_RERATE. See ["Customizing Event Searches for Selective Rerating"](#).

How Comprehensive Rerating Works

To rerate events, use PCM_OP_SUBSCRIPTION_RERATE_REBILL.

This opcode rerates the events for accounts identified by the **pin_rerate** utility, rerating one account at a time. The rerating start time is specified in the input flist. This opcode calls other opcodes to perform rerating functions.

PCM_OP_SUBSCRIPTION_RERATE_REBILL generates either a shadow event or an adjustment event to apply the results of rerating to the account balance in the current billing cycle:

- If the event is unbilled, a shadow event is created, which is an adjustment event that is applied to the bill item instead of the adjustment item.
- If the event is billed, or if the event is unbilled but already posted to the G/L, an adjustment event is created and applied to the adjustment item.

PCM_OP_SUBSCRIPTION_RERATE_REBILL does not rerate, but only reapplies the balance impacts of the original event for these event types: adjustment, payment, writeoff, dispute, settlement, refund, charge, item transfer, cycle fold, cycle tax, reversal, and pre-rated.

If there is no balance impact associated with an event or if there is no rated quantity on the original event, the event is not rerated.

The input to PCM_OP_SUBSCRIPTION_RERATE_REBILL includes:

- The POID of the account to rerate.
- The time from which to start rerating. Events occurring from this start time are rerated.
- Flags that modify rerating behavior. See ["Flags Used for Rerating"](#).
- The session object. When PCM_OP_SUBSCRIPTION_RERATE_REBILL is invoked by the **pin_rerate** utility, the session object passed is the rerating audit object (**/event/control_rerate**). This object is used to generate rerating reports.

PCM_OP_SUBSCRIPTION_RERATE_REBILL does the following:

1. Closes the billing cycle, if due.
2. Retrieves all events for the account from the start date specified.
3. Determines the balance for each resource in the account when rerating starts. This is the current balance minus the cumulative total of all balance impacts for the resource from the rerating start time until the current time.
4. For each event:
 - If the effective time is passed in the PIN_FLD_START_T field, no balance impact is required and the event is not rerated.
 - Checks the **/data/ledger_report** object to determine whether the G/L for the event has been posted.
 - Calls the PCM_OP_ACT_USAGE opcode to determine the difference between the original rated event and the rerated event.

Note: PCM_OP_ACT_USAGE does not check credit limits when rerating events.

- Calls PCM_OP_ACT_USAGE again to apply the results of rerating to the account balance. If the event is sponsored, the balance impacts are applied to the sponsoring account.
- Updates the RERATE_OBJ field for the event being rerated with the POID of the new rerated event.

The account balance is updated after rerating each event so that each subsequent event is rerated based on the most current balance.

Some rerated events may be associated with a product, discount, or resource balance whose validity period starts on first usage. If the event that triggered the validity period is not the actual first usage event, PCM_OP_SUBSCRIPTION_RERATE_REBILL backs out the validity period of the product, discount, or resource balance and resets it based on the event time of the actual first-usage event.

When override pricing is passed to PCM_OP_SUBSCRIPTION_RERATE_REBILL, it creates a */rerate_session/override_info* object to capture the override pricing information used during rerating.

Flags Used for Rerating

The PIN_FLD_FLAGS field specifies whether to rerate events in order based on the event creation time or the event end time. The end time is the default. For more information, see "[Specifying the Event Sequence for Rerating](#)".

The PIN_FLD_RERATE_FLAGS field can take two flags:

- A flag that specifies back-out-only rerating. When this flag is set, the balance impacts of the events selected for rerating are backed out, but the events are not rerated.
- A flag that specifies selective rerating of specific events. When this flag is set, the account selection criteria is also applied to the account's events and only events that meet the selection criteria are rerated.

Important:

- Do not use selective rerating if your rating scheme includes credit limits or resource-based tiers. These schemes require that all events related to an account are rerated to assure accurate rerating.
 - Do not use selective rerating if deferred taxation is used for taxing events during rerating.
 - Use caution when specifying selective rerating. Account balances can be impacted by different types of events and the order of the balance impacts is important to accurately apply discounts and consume resources. It is typically safer to rerate all of an account's events.
-
-

For more information, see "[Specifying Events for Rerating](#)".

The selection criteria for selective rerating and back-out-only rerating must be set in the PIN_FLD_ARGS array field in the input flist.

Return Values for Rerating

If the CALC_ONLY flag is set or if PCM_OP_FLAG_READ_RESULT is passed in the input flist, PCM_OP_SUBSCRIPTION_RERATE_REBILL returns all rerating details in the PIN_FLD_RESULTS array without modifying the account balances. Otherwise, the account balances are updated and only the POID of the rerated event is returned.

If PCM_OP_SUBSCRIPTION_RERATE_REBILL attempts to rerate an event that is already being rerated, an error is returned.

How BRM Creates Rerate Jobs

A rerate job is a pair of `/job/rerate` and `/job_batch/rerate` objects in the BRM database. There is a one-to-one relationship between `/job/rerate` objects and `/job_batch/rerate` objects and both are created in the same transaction when one or more accounts are selected for rerating.

You create rerate jobs manually by using the `pin_rerate` utility. BRM creates rerate jobs automatically by using the PCM_OP_RERATE_INSERT_RERATE_REQUEST opcode. This opcode is called to create rerate jobs by the following:

- The `pin_rerate` utility.
- BRM's automatic-rerating mechanism.
BRM supports the automatic rerating of certain types of events. For more information, see ["About Automatic Rerating"](#).
- Your trigger-dependent rerating configuration.
BRM enables you to rerate events automatically based on your own business requirements. For more information, see ["About Trigger-Dependent Rerating"](#).
- Out-of-order rerating for pipeline rated events.
BRM supports the automatic rerating of events that are rated out of order in the batch rating pipeline. For more information, see ["About Automatic Rerating of Out-of-Order Events"](#).

PCM_OP_RERATE_INSERT_RERATE_REQUEST receives the following information in the input flist:

- The POIDs of the accounts to be rerated (required).
- The time from which events must be rerated for the accounts (required).
- Accounts related to the accounts to be rerated (for example, an account that used the account's service before a line transfer) along with the start time for each account.

Note: If more than one account is included in the rerate job, the same rerate start time, selection criteria, and price overrides apply to all the accounts.

- Your rerate reason for trigger-dependent rerating (the default rerate reason passed in for BRM automatic rerating scenarios is 0).

- Your selection criteria for trigger-dependent rerating to identify the events that must be rerated for the accounts.
- A list of products, discounts, or deals to override the subscriber's existing pricing objects during rerating.

To create rerate jobs, PCM_OP_RERATE_INSERT_RERATE_REQUEST does the following:

1. Checks for duplicate rerate job requests.

For more information, see ["How BRM Handles Duplicate Rerate Jobs"](#).

2. Calls PCM_OP_RERATE_CREATE_JOB to create the `/job/rerate` and `/job_bacth/rerate` objects.

How BRM Handles Duplicate Rerate Jobs

The PCM_OP_RERATE_INSERT_RERATE_REQUEST opcode checks for existing jobs when a new rerate request is made; this opcode compares the data for each top-level account in the rerate request to the data in the existing jobs. To avoid duplicate rerate jobs, accounts are either eliminated from or updated in the rerate request or the existing jobs.

After the duplicate detection process is complete for each top-level account in the rerate request, PCM_OP_RERATE_INSERT_RERATE_REQUEST calls PCM_OP_RERATE_CREATE_JOB to create a rerate job with the resulting contents of the rerate request. PCM_OP_RERATE_CREATE_JOB then performs checks on the sub-accounts in the rerate request to check for duplicate jobs for line transfers before creating the rerate job.

Detecting Duplicate Rerate Requests

When the rerate reason code and rerate selection criteria for a rerate request match those of one or more rerate jobs, the request is considered a possible duplicate. When the rerate reason code and rerate selection criteria do not match, the rerate request is not a duplicate, and a new rerate job is created.

Avoiding Duplication of Rerate Jobs

To avoid duplicate rerate jobs for an account, PCM_OP_RERATE_INSERT_RERATE_REQUEST compares the following values for duplicate rerate job requests:

- The rerate start times. BRM always uses the earlier start time to ensure that all events are rated.
- The price overrides (if any). BRM uses the latest price overrides for an account because it is assumed to be the most updated.

When price overrides match

When the price overrides match, PCM_OP_RERATE_INSERT_RERATE_REQUEST does the following for each top-level account:

- If the rerate start time of the rerate request is later than or equal to the start time of the existing job, does the following:
 - If there is only this account in the rerate request, deletes the request. No rerate job is needed.
 - If there are many accounts in the new request, removes only that account from the new request.

- If the rerate start time of the rerate request is earlier than the start time of the existing job, does the following:
 - If the existing job contains only this account, updates the existing job to use the earlier start time and removes the account from the rerate request.
 - If the existing job contains other accounts, removes the account from the existing job, keeping the account in the rerate request.

When price overrides do not match

When the price overrides do not match, PCM_OP_RERATE_INSERT_RERATE_REQUEST does the following for each top-level account:

- If the rerate start time of the rerate request is the same or earlier than the start time of the existing job, does the following:
 - If there is only one account in the existing job, deletes the job.
 - If there are many accounts in the existing job, keeps the account in the rerate request so the new rerate job start time and price overrides are used. Removes the account from the existing job.
- If the rerate start time of the rerate request is later than the start time of the existing job, does the following:
 - If there is only one account in the existing job, deletes the job.
 - If there are many accounts in the existing job, keeps the account in the rerate request so the new price overrides are used. Updates the rerate request start time to use the existing job’s start time. Removes the account from the existing job.

Table 28–4 summarizes how PCM_OP_RERATE_INSERT_RERATE_REQUEST handles duplicate rerate job requests:

Table 28–4 Request Handling by PCM_OP_RERATE_INSERT_RERATE_REQUEST

If the Start Time of the New Request Is:	And the Price Overrides:	Perform This Action:
Equal to or later than that of the existing job	Match	If there is only this account in the rerate request, delete the request. No rerate job is needed. If there are many accounts in the rerate request, remove only that account from the rerate request.
Earlier than that of the existing job	Match	If there is only one account in the existing job: <ul style="list-style-type: none"> ■ Update the existing job to use the earlier start time. ■ Remove the account from the rerate request. If there are many accounts in the existing job: <ul style="list-style-type: none"> ■ Keep the account in the rerate request. ■ Remove the account from the existing job.
Later than that of the existing job	Do not match	If there is only one account in the existing job, delete the job. If there are many accounts in the existing job: <ul style="list-style-type: none"> ■ Keep the account in the rerate request so the new price overrides are used. ■ Update the rerate request job start time to use the existing job’s start time. ■ Remove the account from the existing job.

Table 28–4 (Cont.) Request Handling by PCM_OP_RERATE_INSERT_RERATE_

If the Start Time of the New Request Is:	And the Price Overrides:	Perform This Action:
Equal to or earlier than that of the existing job	Do not match	If there is only one account in the existing job, delete the existing job. If there are many accounts in the existing job: <ul style="list-style-type: none"> ■ Keep the account in the rerate request so the rerate request start time and price overrides are used. ■ Remove the account from the existing job.

For more information on how BRM creates rerate jobs, see ["How BRM Creates Rerate Jobs"](#).

Tracking Rate Changes for Rerating

To track rate changes for rerating, use the PCM_OP_SUBSCRIPTION_PREP_RATE_CHANGE opcode.

This opcode creates the `/rate_change` object, which stores details about the products affected by a rate change, including the rate tiers and rate plans for the product.

When you change a cycle fee by adding a new rate tier in Pricing Center, BRM event notification triggers this opcode. This opcode reads the products associated with the event and creates a `/rate_change` object, which is used by the `pin_rate_change` utility to create rerating requests.

PCM_OP_SUBSCRIPTION_PREP_RATE_CHANGE returns the POID of the `/rate_change` object.

Rerating Cycle Fees

To rerate cycle fees, use the PCM_OP_SUBSCRIPTION_RATE_CHANGE opcode. This opcode uses the event notification mechanism to trigger the creation of rerating requests when there is a cycle-forward or cycle-forward-arrears event rate change in the middle of the current cycle.

Note: Rerating is not triggered for cycle-arrears rate changes or rate changes in future cycles.

When you run the `pin_rate_change` utility after a rate change, the utility calls this opcode and provides details about the accounts and products affected by the rate change.

This opcode returns a notification event of type `/event/notification/rate_change` for each account picked up by the `pin_rate_change` utility. Depending on how automatic rerating is configured, the notification event triggers the creation of rerating requests.

Configuring Comprehensive Rerating

This chapter describes how to configure your Oracle Communications Billing and Revenue Management (BRM) system for comprehensive rerating using `pin_rerate`.

Note: This documentation does not apply to rerating only pipeline-batch-rated events by using a batch rerating pipeline.

For an overview of comprehensive rerating, see "[About Comprehensive Rerating Using `pin_rerate`](#)".

Before reading this chapter, you should be familiar with the Pipeline Manager architecture. See *BRM Configuring Pipeline Rating and Discounting*.

About Configuring Comprehensive Rerating

How you configure rerating depends on whether you rerate real-time-rated and pipeline-batch-rated events concurrently or, if you do not use Pipeline Manager for batch rating, rerate only real-time-rated events. See one of the following sections:

- [Configuring Concurrent Rerating of Pipeline-Rated and Real-Time-Rated Events](#)
- [Configuring Rerating When You Do Not Use a Batch Rating Pipeline](#)

You perform the following configurations for both concurrent rerating and real-time-event only rerating:

- [Specifying Whether the Batch Rating Pipeline Is Enabled](#)
- (Optional) "[Setting the Rerating Event Cache Size \(Fetch Size\)](#)"
- (Optional) "[Configuring the Number of Accounts Per Job and Number of Jobs per Transaction](#)"

You can also configure BRM to automatically create rerate jobs when the following events occur:

- When rates are changed. See "[About Calculating Charges When You Change the Rate](#)" in *BRM Configuring and Running Billing*.
- When rollover corrections are made due to delayed events. See "[Enabling Rerating and Rollover Correction Due to Delayed Events](#)" in *BRM Configuring and Running Billing*.
- When certain events, such as purchase and cancellation events, are backdated. See "[About Automatic Rerating of Backdated Events](#)".

- When pipeline batch-rated events are rated out of order. See ["About Automatic Rerating of Out-of-Order Events"](#).
- When events trigger rerating based on your custom rerating requirements. See ["About Trigger-Dependent Rerating"](#).

Configuring Concurrent Rerating of Pipeline-Rated and Real-Time-Rated Events

Important: Before you can configure concurrent rerating, you must have installed and configured Pipeline Manager and the Account Synchronization Data Manager (DM).

To configure concurrent rerating of both pipeline-rated and real-time-rated events, perform these tasks:

- Configure a real-time rerating pipeline. See ["Configuring a Real-Time Rerating Pipeline"](#).
- Configure **pin_rerate** and the batch rating pipeline. See ["Configuring the Batch Rating Pipeline and pin_rerate to Synchronize Processing"](#).
- (Optional) Configure the rerating event cache size. See ["Setting the Rerating Event Cache Size \(Fetch Size\)"](#).
- (Optional) Configure parameters for creating rerate jobs. See ["Configuring the Number of Accounts Per Job and Number of Jobs per Transaction"](#).

Configuring a Real-Time Rerating Pipeline

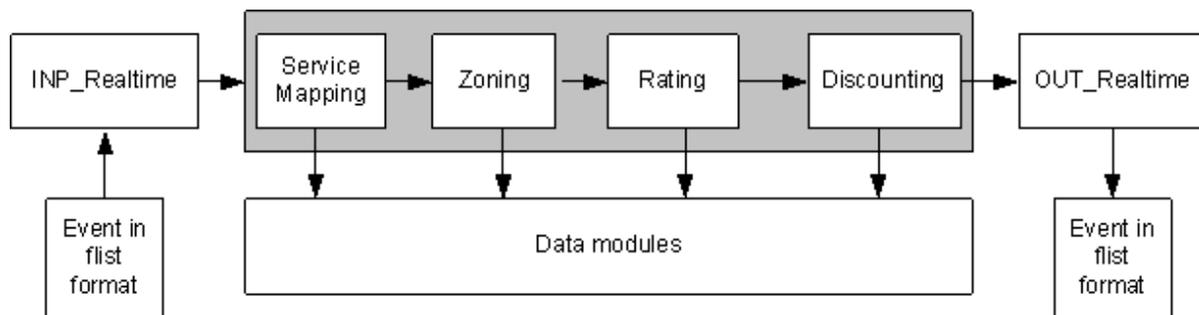
A real-time rerating pipeline is configured similarly to a batch rerating pipeline, but with fewer preprocessing and post-rating modules. This is because most of the enrichment data is provided in the input flist received from the CM.

You typically create a separate instance of Pipeline Manager for real-time rerating. The default registry file is *BRM_Home/conf/wirelessRealtime.reg*.

To configure a real-time rerating pipeline, perform the following tasks:

- (Optional) ["Configuring Multiple Real-Time Rerating Pipelines"](#)
- [Configuring the Real-Time Rerating Data Pool](#)
- [Configuring the Modules in the Real-Time Rerating Pipeline](#)
- [Adding Real-Time Rerating Pipeline Data to the IFW_PIPELINE Table](#)

Figure 29–1 shows the real-time rerating pipeline architecture:

Figure 29–1 Real-Time Rerating Pipeline Architecture

Configuring Multiple Real-Time Rerating Pipelines

To improve performance or scalability, you configure multiple instances of real-time rerating pipelines. For example, you can increase performance by configuring multiple pipelines to process rerate requests in parallel or by configuring multiple real-time rerating pipelines to process different rerate requests: for example, by configuring one pipeline that rerates only GPRS events and another that rerates only GSM events. See "Configuring Multiple Instances of a Pipeline" in *BRM System Administrator's Guide*.

Configuring the Real-Time Rerating Data Pool

Configure the following data modules:

- DAT_Zone
- Rateplan
- DAT_Calendar
- DAT_TimeModel
- DAT_PriceModel
- Dayrate
- DAT_Currency
- DAT_USC_Map

See "Configuring the Data Pool" in *BRM System Administrator's Guide*

Configuring the Modules in the Real-Time Rerating Pipeline

Configure the following modules in the real-time rerating pipelines:

- Configure the INP_Realttime input module. Specify the flist-to-EDR mapping file used for rerating in the **OpcodeMapping** registry entry. For example:
OpcodeMapping = ./formatDesc/Formats/Realtime/rate_event.xml
- Configure the NET_EM module:
 - Configure NET_EM to manage data between the CM and Pipeline Manager and configure the CM to send rerate request to the NET_EM module.
 - Configure NET_EM to route rerate requests. See "[Configuring NET_EM to Route Rerate Requests Based on the Event Field Value](#)".
- Configure the output module to add any customizations to the output flist.
- Configure these function modules:

- FCT_ServiceCodeMap
- FCT_CustomerRating
- FCT_PreRating
- FCT_IRules
- FCT_USC_Map
- FCT_RSC_Map
- FCT_MainRating
- FCT_Dayrate
- FCT_RateAdjust
- FCT_Rounding
- FCT_DiscountAnalysis
- FCT_Discount

Configuring NET_EM to Route Rerate Requests Based on the Event Field Value

To configure NET_EM to route rerate requests to multiple real-time rerating pipelines based on the type of event, you set the **FieldName** and **FieldValue** NET_EM module registry entries.

Important: If you use event routing based on the event field value, ensure that the input events contain the expected field name and field values specified in the NET_EM module registry. Otherwise, NET_EM will not be able to route the events.

By using the “.” notation, you can specify a field at any level in the input event flist to be used to route the event. For example, this substruct and field:

```
PIN_FLD_EVENT
  PIN_FLD_POID
```

is represented like this:

```
PIN_FLD_EVENT.PIN_FLD_POID
```

In the NET_EM registry below, if PIN_FLD_EVENT.PIN_FLD_POID is a GSM event, the rerate request is routed to any one of the two instances of the GSM rerating pipeline (RealtimeReratingGSM). If the event is a GPRS event, the rerate request is routed to any one of the two instances of the GPRS rerating pipeline (RealtimeReratingGPRS).

```
DataPool
{
  RealtimePipeline
  {
    ModuleName = NET_EM
    Module
    {
      ThreadPool
      {
        Port = 14579
        Threads = 4
      }
    }
  }
}
```

```

ReratingOpcode
{
  OpcodeName = PCM_OP_RATE_PIPELINE_EVENT
  FieldName = PIN_FLD_EVENT.PIN_FLD_POID
  GSMEvent
  {
    FieldValue = /event/delayed/session/telco/gsm
    PipelineName = RealtimeReratingGSM
    NumberOfRTPipelines = 2
  }
  GPRSEvent
  {
    FieldValue = /event/delayed/session/gprs
    PipelineName = RealtimeReratingGPRS
    NumberOfRTPipelines = 2
  }
}
}
}
}

```

Adding Real-Time Rerating Pipeline Data to the IFW_PIPELINE Table

Pipeline Manager stores information about pipelines in the IFW_PIPELINE table. The pipelines that are preconfigured in the **Pipelines** section of the default registry file (*Pipeline_home/conf/wirelessRealtime.reg*) are inserted into the IFW_PIPELINE table during Pipeline Manager installation.

Important:

- If you are *not* using the default registry file, and you have configured new real-time rerating pipelines, you must manually insert the pipelines into the IFW_PIPELINE table by using SQL commands. Otherwise, you will receive an error at system startup.
 - If you *are* using the default registry file and have changed the default pipeline names or you have configured additional pipelines, you must manually insert the new pipelines into the IFW_PIPELINE table.
-
-

The following example shows SQL commands to insert RealtimeReratingGSM and RealtimeReratingGPRS pipelines into the IFW_PIPELINE table:

```

% sqlplus pin/password@databaseAlias

SQL>INSERT INTO IFW_PIPELINE ( PIPELINE, NAME, EDRC_DESC ) VALUES (
'RealtimeReratingGSM', 'GSM Realtime Rerating Pipeline', 'ALL_RATE');

SQL>INSERT INTO IFW_PIPELINE ( PIPELINE, NAME, EDRC_DESC ) VALUES (
'RealtimeReratingGPRS', 'GPRS Realtime Rerating Pipeline', 'ALL_RATE');

SQL>commit;

```

Configuring the Batch Rating Pipeline and pin_rerate to Synchronize Processing

To enable **pin_rerate** and the batch rating pipeline to synchronize processes, perform the following tasks:

- Ensure that **pin_rerate** can communicate with a batch pipeline by setting the batch rating pipeline business parameter to **enabled**. See "[Specifying Whether the Batch Rating Pipeline Is Enabled](#)".
- Configure rerating business events and event notification. See "[Configuring Rerating Business Events and Event Notification](#)".
- Configure Pipeline Manager to send and receive events. See "[Configuring Pipeline Manager to Handle Business Events from pin_rerate](#)".
- Configure **pin_rerate** to send and receive events. See "[Configuring pin_rerate to Receive Acknowledgment Events from Pipeline Manager](#)".
- Configure standard recycling to enable Pipeline Manager to suspend and recycle EDRs. See "[Configuring Standard Recycling](#)".

Configuring Rerating Business Events and Event Notification

The **pin_rerate** utility and the batch rating pipeline synchronize processing by passing business events by way of the Account Synchronization DM. To generate business events, **pin_rerate** uses event notification.

The business events and notification events used for rerating are defined in configuration files provided with the Account Synchronization DM:

- The notification events used for rerating are specified by the following entries in the Account Synchronization DM notification list (*BRM_Home/sys/data/config/pin_notify_ifw_sync*):


```
1301    0    /event/notification/rerating/start
1301    0    /event/notification/rerating/end
1301    0    /event/notification/rerating/PrepareToRerate
1301    0    /event/notification/rerating/ReratingCompleted
```
- The business events used for rerating are specified in the Account Synchronization DM EAI payload configuration file (*BRM_Home/sys/eai_js/payloadconfig_ifw_sync.xml*).

You configure these files when you install and configure the Account Synchronization DM. See *BRM Synchronization Queue Manager*.

Configuring Pipeline Manager to Handle Business Events from pin_rerate

To enable **pin_rerate** and Pipeline Manager to send and receive business events through the Account Synchronization DM, perform these tasks:

- [Creating the Acknowledgment Queue](#)
- [Configuring Access to Queues in a Multischema System](#)
- [Configuring the DAT_Listener Module](#)

Creating the Acknowledgment Queue

The Account Synchronization DM uses database queues to send business events to Pipeline Manager. A default database queue is created for this purpose when you install the Account Synchronization DM.

You must create an additional database queue for Pipeline Manager to post acknowledgment events that are dequeued and processed by **pin_rerate**.

Note: Only one business event queue and one acknowledgment queue is required for each instance of Pipeline Manager in your system. You do not need to create additional acknowledgment queues if one has already been created.

To create an acknowledgment queue, use the `pin_ifw_sync_oracle.pl` utility. For example:

```
pin_ifw_sync_oracle.pl create -q ACK_QUEUE -t ack_queue_t -l
user/password@database
```

Configuring Access to Queues in a Multischema System

You can run BRM and Pipeline Manager in a multischema environment. For example, you might have a schema for BRM that uses the **PIN** logon, a schema for Account Synchronization queue that uses the **PINQ** logon, and a schema for the instance of Pipeline Manager that uses the **INTEGRATE** logon.

In such cases, you must create a global synonym for the Account Synchronization acknowledgment queue. This enables the user of the BRM schema to access the acknowledgment queue in the Account Synchronization schema.

1. Using SQL*Plus, log in to your database as the SYSTEM user and create a global user synonym:

```
sqlplus system/manager@DatabaseAlias
```

```
CREATE PUBLIC SYNONYM ACCT_SYNC for PINQ.ACCT_SYNC
```

where PINQ is the login for the schema that includes the Account Synchronization stored procedures and acknowledgment queue.

Note: The rerating stored procedures run in the PIN schema (the BRM schema).

2. Add the global synonym to the `pin_rerate` configuration file (`BRM_Home/app/pin_rerate/pin.conf`):

```
- pin_rerate ack_queue PINQ.ACK_QUEUE
```

Configuring the DAT_Listener Module

Pipeline Manager responds to business events sent by `pin_rerate` by posting acknowledgment events. You must configure the DAT_Listener module to post acknowledgment events to the Account Synchronization acknowledgment queue. Set the `AckQueueName` entry to specify the name of the acknowledgment queue.

Sample DAT_Listener registry:

```
Listener
{
  ModuleName = DAT_Listener
  Module
  {
    InfranetConnection = ifw.DataPool.LoginQueue
    QueueName = QUEPIN17
    AckQueueName = ACK_QUEUE
    MQAckServerName = ACK_QUEUE_SERVER
```

```
        LogEvents = TRUE  
    }  
}
```

Configuring `pin_rerate` to Receive Acknowledgment Events from Pipeline Manager

To enable `pin_rerate` to receive acknowledgment events from Pipeline Manager, perform the following tasks:

- [Loading the Stored Procedure for Rerating](#)
- [Configuring `pin_rerate` to Dequeue Events from the Acknowledgment Queue](#)

Loading the Stored Procedure for Rerating

The `pin_rerate` utility uses stored procedures to dequeue acknowledgment events from the database queue and to purge rerate jobs (for more information on purging rerate jobs, see "[Parameter for Purging Rerate Jobs](#)").

Note: Before loading the `pin_rerate` stored procedures, you must have Account Synchronization installed and configured.

To load the stored procedure:

Oracle:

On Oracle systems, you manually load the rerating stored procedures `job_rerate_procedures_pkb.plb` and `job_rerate_procedures_oracle.plb` into your BRM database:

1. Connect to your database using SQL*Plus:

```
sqlplus user/password@database
```

where *user* and *password* are your user name and password, and *database* is the service name for the BRM database.

2. At the SQL*Plus prompt, enter the following commands:

```
@BRM_Home/sys/dm_oracle/data/job_rerate_procedures_pkg_oracle.plb  
@BRM_Home/sys/dm_oracle/data/job_rerate_procedures_oracle.plb
```

where *BRM_Home* is the directory in which BRM is installed.

Configuring `pin_rerate` to Dequeue Events from the Acknowledgment Queue

Configure the `pin_rerate` utility to dequeue events from the acknowledgment queue that you created in "[Creating the Acknowledgment Queue](#)".

Add the following entry in the `pin_rerate` configuration file (*BRM_Home/app/pin_rerate/pin.conf*):

```
- pin_rerate ack_queue ACK_QUEUE
```

where `ACK_QUEUE` is the name of the acknowledgment queue.

Configuring Standard Recycling

BRM rerating uses standard recycling for two purposes:

- To load EDRs suspended during rerating into the BRM database.
- To retrieve suspended EDRs from the BRM database and recycle them when rerating is complete.

For configuration instructions, see "Configuring Standard Recycling" in *BRM Configuring Pipeline Rating and Discounting*.

Configuring Rerating When You Do Not Use a Batch Rating Pipeline

If you do not use Pipeline Manager for batch rating, perform the following tasks to configure `pin_rerate` to rerate only real-time-rated events:

- Ensure that `pin_rerate` does not attempt to communicate with a batch rating pipeline by setting the batch rating pipeline business parameter to **disabled**. See "Specifying Whether the Batch Rating Pipeline Is Enabled".
- (Optional) Configure the rerating event cache size. See "Setting the Rerating Event Cache Size (Fetch Size)".
- (Optional) Configure parameters for creating rerate jobs. See "Configuring the Number of Accounts Per Job and Number of Jobs per Transaction".

Specifying Whether the Batch Rating Pipeline Is Enabled

When the batch pipeline is enabled, `pin_rerate` generates notification events to synchronize processing with the batch pipeline by way of the Account Synchronization DM. When the batch pipeline is disabled, `pin_rerate` does not generate notification events because synchronization is not needed.

`pin_rerate` checks a field in the `rerate` instance of the `/config/business_params` object, to determine if the batch rating pipeline is enabled or disabled:

- To rerate both real-time-rated and pipeline-rated events concurrently, the batch rating pipeline entry must be set to **enabled**.
- To rerate only real-time-rated events, the batch rating pipeline entry must be set to **disabled**.

The default is **enabled**.

You modify the `/config/business_params` object by using the `pin_bus_params` utility.

To specify whether the batch pipeline is enabled or disabled:

1. Use this command to create an editable XML file from the `rerate` instance of the `/config/business_params` object:

```
pin_bus_params -r BusParamsRerate bus_params_rerate.xml
```

This command creates the XML file named `bus_params_rerate.xml.out` in your working directory. If you do not want this file in your working directory, specify the full path as part of the file name.

2. Search the XML file for following line:

```
<BatchRatingPipeline>enabled</BatchRatingPipeline>
```

3. Set the entry as needed:

- **enabled**: To rerate both real-time-rated and pipeline-rated events concurrently.
- **disabled**: To rerate only real-time-rated events when you do not use Pipeline Manager for batch rating.

Caution: BRM uses the XML in this file to overwrite the existing **rerate** instance of the **/config/business_params** object. If you delete or modify any other parameters in the file, these changes affect the associated aspects of the BRM billing configuration.

4. Save the file and change the file name from **bus_params_rerate.xml.out** to **bus_params_rerate.xml**.
5. Use the following command to load the change into the **/config/business_params** object:

```
pin_bus_params bus_params_rerate.xml
```

You should execute this command from the *BRM_Home/sys/data/config* directory, which includes support files used by the utility.
6. Read the object with the **testnap** utility or the Object Browser to verify that all fields are correct.
7. Stop and restart the Connection Manager (CM).
8. (Multischema systems only) Run the **pin_multidb** script with the **-R CONFIG** parameter. For more information, see "pin_multidb" in *BRM System Administrator's Guide*.

Setting the Rerating Event Cache Size (Fetch Size)

By default, BRM rerating caches 10,000 events in system memory for processing. Depending on your system memory, you can set the **event_fetch_size** in the Connection Manager's configuration file to specify the number of events retrieved from the database and cached in the system memory for processing.

To set the event cache size:

1. Open the Connection Manager (CM) configuration file (*BRM_Home/sys/cm/pin.conf*) using a text editor.
2. Uncomment the **- fm_subscription event_fetch_size** entry.
3. Edit the **event_fetch_size** value. The default is 10000.

```
- fm_subscription event_fetch_size 10000
```
4. Save the file.
5. Stop and restart the CM.

Configuring the Number of Accounts Per Job and Number of Jobs per Transaction

By default, BRM assigns 10 accounts to each rerate job and creates 2 rerate jobs per transaction. For example, if the total number of accounts to rerate is 10,000, BRM creates 1000 rerate jobs. It creates the rerate jobs in 500 separate transactions: 2 jobs in each transaction.

To change the default number of accounts per job and the number of rerate jobs created per transaction, perform the following tasks:

1. Open the **pin_rerate** configuration file (*BRM_Home/apps/pin_rerate/pin.conf*).

2. Set the number of accounts assigned to each rerate job by adding the following line:

```
- pin_rerate_per_job accounts_per_job
```

where *accounts_per_job* is the number of accounts to assign to each job.

3. Set the number of jobs created per transaction by adding the following line:

```
- pin_rerate_per_transaction jobs_per_transaction
```

where *jobs_per_transaction* is the number of jobs to create in each transaction.

4. Save and close the file.

Caution: Setting the **pin_rerate_per_job** entry to a small number, for example 1, will result in many rerate jobs being created. Too many rerate jobs can affect your system's performance due to the rerate steps performed for each rerate job. Processing multiple accounts in one rerate job reduces the total number of rerate steps performed compared to processing those same accounts in multiple rerate jobs.

Configuring Rerating to Reset First-Usage Validity Periods

Perform this task if your price plan includes products, discounts, or balance impacts that become valid on first usage (when customers use the products and discounts or consume the resource balance for the first time).

For information about first-usage validity, see "[About Effective Periods That Start on First Usage](#)" and "[About Balance Impacts That Become Valid on First Usage](#)".

When rerated events are associated with products, discounts, or resource balances that start on first usage, rerating resets their validity periods, if necessary. For example, if the first event rated, which initiated a product's validity period, was not actually the first event to use the product's service, rerating corrects the order of the events and resets the validity period based on the actual first-usage event.

To configure rerating for first-usage validity, perform the tasks in the following sections:

- [Configuring the Real-Time Rerating Pipeline to Set Product Validity Periods.](#)
- [Configuring Event Notification for Rerating Cycle Events Triggered by First Usage.](#)

Configuring the Real-Time Rerating Pipeline to Set Product Validity Periods

If your products are configured to start when they are first used, configure the `ISC_FirstProductRealtime` iScript in the real-time rerating pipeline.

When products start on first usage, the real-time rerating pipeline adds the account's first-usage product and discount information to the EDR. `ISC_FirstProductRealtime` sets the validity period in the BRM database for products that were used to rate the event and that start on first usage. This triggers any purchase and cycle fees.

Configuring Event Notification for Rerating Cycle Events Triggered by First Usage

To ensure that BRM can rerate cycle events that are triggered because of first usage, you must add cycle-event notification events to your event notification list (the default notification list is `BRM_Home/sys/data/config/pin_notify`). BRM provides a list of

default cycle-event notification events in the **pin_notify_rerate** file. Use this file to update your **pin_notify** file.

To configure event notification for rerating cycle events triggered by first usage:

1. Open the **pin_notify** and **pin_notify_rerate** files in *BRM_Home/sys/data/config*.
2. Copy the entries in the **pin_notify_rerate** file to the end of the **pin_notify** file.
3. Save and close the **pin_notify** file and close the **pin_notify_rerate** file.
4. Load the contents of the **pin_notify** file into the **/config/notify** object in the BRM database by running the **load_pin_notify** utility.

For more information, see "Using Event Notification" in *BRM Developer's Guide*.

Configuring Rerating for Accounts Associated With Subscription Service Transfer

When a subscription service is transferred from one subscriber account to another, rerating the original account can affect the subscription service balances transferred to the new account. However, by default, the rerating process only selects accounts for rerating based on your specified search criteria. This means that, when an account selected for rerating is associated with a subscription service transfer, the other accounts to which the subscription service was transferred are not selected for rerating.

To rerate all accounts associated with a subscription service transfer, you configure the **LineManagement** business configuration parameter. When this parameter is enabled, the rerating process searches for any subscription service transfers for the account that is rerated, and adds the accounts to which the subscription service was transferred during the rerating time specified to the rerate request.

For example:

1. Subscription service X is originally owned by Account A and transferred to Account B on June 15.
2. On June 20, the service is transferred again from Account B to Account C.
3. On July 10, Account A is selected for rerating and the rerating time specified is June 1. Rerating also selects Accounts B and C because the subscription service transfer to Account B and C occurred after June 1. Accounts A, B, and C are grouped together to form a single rerate request.
4. Account A is rerated from June 1 to July 10, Account B is rerated from June 15 to July 10, and Account C is rerated from June 20 to July 10. Rerating selects the events for accounts A, B, and C, and rerates the events in chronological order, resulting in correct subscription service balance updates.

Note: If rerating fails for any one of the accounts in the rerate request, rerating fails for all the accounts in the rerate request. For more information, see "[How Failed Rerate Jobs Are Processed](#)".

During rerating, each account is locked only for as long as it takes to rerate its events. However, in case of a subscription service transfer, related accounts are locked for the duration of rerating events associated with all of those accounts. In the example above, accounts A, B, and C remain locked until rerating of all three accounts is complete.

By default, the **LineManagement** parameter is disabled and cached at CM startup. To change the default value, you modify the **rerate** instance of the **/config/business_params** object by using the **pin_bus_params** utility. For information on this utility, see "pin_bus_params" in *BRM Developer's Guide*.

To enable rerating of accounts associated with a subscription service transfer:

1. Run this command to create an editable XML file from the **rerate** instance of the **/config/business_params** object:

```
pin_bus_params -r BusParamsRerate bus_params_rerate.xml
```

This command creates the XML file named **bus_params_rerate.xml.out** in your working directory. If you do not want this file in your working directory, specify the full path as part of the file name.

2. Search the XML file for following line:

```
<LineManagement>0</LineManagement>
```

3. Change **0** to **1**.

Caution! BRM uses the XML in this file to overwrite the existing **rerate** 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 BRM's billing configuration.

4. Save the file and change the file name from **bus_params_rerate.xml.out** to **bus_params_rerate.xml**.
5. Go to the *BRM_Home/sys/data/config* directory.
6. Run the command:

```
pin_bus_params PathToWorkingDirectory/bus_params_rerate.xml
```

where *PathToWorkingDirectory* is the directory in which **bus_params_rerate.xml** resides.

Note: To run this command from a different directory, see the description for **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.

See "Using testnap" in *BRM Developer's Guide* for general instructions on using testnap.

See "Reading objects by using Object Browser" in *BRM Developer's Guide* for information on how to use Object Browser.

8. Stop and restart the Connection Manager (CM). For more information, see "Starting and stopping the BRM system" in *BRM System Administrator's Guide*.
9. (Multischema systems only) Run the **pin_multidb** script with the **-R CONFIG** parameter. For more information, see "pin_multidb" in *BRM System Administrator's Guide*.

About Automatic Rerating of Backdated Events

Note: Automatic rerating *does not* mean that rerating occurs immediately when an event is backdated. Automatic rerating means that rerate jobs (`/job/rrerate` and `/job_rrerate/rrerate` objects) are automatically created.

When backdated events occur, BRM uses event notification to automatically create rerate jobs to rerate the backdated events. The accounts in the rerate jobs are rerated the next time you run "`pin_rrerate`" to process rerate jobs; you do not need to specify the accounts and events in the `pin_rrerate` command line. For more information about `pin_rrerate`, see "[Using the pin_rrerate Utility](#)".

BRM supports automatic rerating of backdated events in the following cases:

- When purchase or cancellation of a product, discount, or deal is backdated.
- When adjustment to a non-currency resource is backdated.
- When an extended rating attribute (ERA) modification is backdated.

[Table 29–1](#) shows the entries in the CM configuration file used to create rerate jobs for backdated events:

Table 29–1 Configuration File Entries for Rerate Jobs

Entry	Description
<code>backdate_trigger_auto_rrerate</code>	Specifies whether automatic rerating is enabled.
<code>backdate_window</code>	Specifies the minimum time difference needed between the current time and the backdated event time for triggering automatic rerating.
<code>num_billing_cycles</code>	Specifies the maximum number of billing cycles allowed between the current time and the backdated event time of a backdated operation.

Important: All of these conditions must be met to trigger automatic rerating. Otherwise, the backdated event is not rerated.

By default, BRM creates rerate jobs for backdated events when the following conditions are met:

- Automatic rerating is enabled.
- The backdated event time is at least one hour earlier than the current time.
- The backdated event date is not older than one billing cycle.

You can change the default backdated time and date thresholds in the CM configuration file. See "[Configuring Automatic Rerating of Backdated Events](#)".

You can backdate beyond the number of billing cycles specified in the `num_billing_cycles` entry without requesting to automatically rerate. For more information, see "[Backdating beyond Configured Billing Cycles without Automatic Rerating Request](#)".

For more information about using `pin_rrerate`, see "[pin_rrerate](#)".

About Backdated Deal, Product, and Discount Purchase

Automatic rerating of backdated deal, product, and discount purchases is triggered when the difference between the *current time* and the backdated event *purchase, cycle*, or the *usage start time* is greater than the time specified by the **backdate_window** parameter.

BRM then validates that the backdated purchase occurred within the number of billing cycles specified by the **num_billing_cycles** parameter. If the validation is successful, the account is automatically rerated the next time you run **pin_rerate**.

For example, a subscriber's billing day is on the first day of the month. The subscriber purchases a product on July 19 at 5:00 a.m. The CSR backdates the purchase start date to July 15. If the **backdate_window** is set to **1** hour and **num_billing_cycles** is set to **1** cycle, BRM validates that the purchase start time is on or before 4:00 a.m., July 19, and that the purchase start date is not earlier than June 1 (one billing cycle). Because both conditions are met and automatic rerating is enabled, when **pin_rerate** is run with **-rerate** parameter on July 20, BRM automatically rerates all the account's events that occurred after midnight of July 14 to July 20.

About Backdated Deal, Product, and Discount Cancellation

Automatic rerating of backdated deal, product, and discount cancellation is triggered when the difference between the *current time* and the *purchase, cycle*, or the *usage end time* is greater than the time specified by the **backdate_window** parameter.

BRM then validates that the backdated cancel occurred within the number of billing cycles specified by the **num_billing_cycles** parameter. If the validation is successful, the account is automatically rerated the next time you run **pin_rerate**.

The following example demonstrates how fees are prorated and charges are reversed when a backdate product cancellation occurs.

In this example, the backdate parameters are as follows:

- **backdate_window** is 2 hours.
- **num_billing_cycles** is 2 cycles.

The subscriber's product includes the following:

- IP product purchase fee: \$10
- IP product monthly cycle forward fee: \$20 (with proration enabled)
- IP product cancellation fee: \$50
- IP usage fee: \$1 per minute
- Email product monthly cycle forward fee: \$8

1. On January 1, the subscriber purchases the product.

Account balance = \$10 IP product purchase fee + \$20 IP monthly cycle forward fee + \$8 email cycle forward fee = \$38

2. On January 20, the subscriber generates a usage of \$10 for the IP service.

Account balance = \$38 + \$10 usage = \$48

3. On February 1, when billing is run:

Account balance = \$48 + \$20 IP monthly cycle forward fee (for February) + \$8 email monthly cycle forward fee (for February) = \$76

4. On February 10 at 5:00 p.m., the subscriber cancels the IP product. The CSR backdates the product cancellation to January 10. BRM automatically creates a rerate job and the backdate cancel event is rerated by running **pin_rerate**. After backdate cancellation of the IP Product:
 - A \$50 IP product cancellation fee is applied.
 - The IP product monthly cycle forward fee for January is prorated and charged for 10 days only.
 - The \$10 IP service usage fee is reversed.
 - The IP product monthly cycle forward fee for February is reversed.

Account balance = \$10 IP purchase fee + \$20 IP monthly cycle forward fee (for January) - \$14.19 prorated IP monthly cycle forward fee refund + \$8 email monthly cycle forward fee (for January) + \$50 IP product cancellation fee + \$8 email cycle forward fee (for February) = \$81.81

5. On March 1, when billing is run:

Account balance = \$81.81 + \$8 email cycle forward fee (for March) = \$89.81

BRM validates that the purchase end time is on or before 3:00 p.m., February 10, and that the purchase end date is not earlier than December 1 (two billing cycles). When both conditions are met and automatic rerating is enabled, when "**pin_rerate**" is run, BRM automatically rerates all the account's events that occurred after midnight January 10 to February 10.

About Backdated Adjustment of Non-Currency Resources

When you backdate an adjustment of a non-currency resource and automatic rerating is enabled, BRM automatically rerates all the associated events the next time you run **pin_rerate** with the **-rerate** parameter.

Note:

- The adjustment event is rerated only if the adjustment is for a non-currency resource.
 - The adjustment must be backdated so that it can be used to rerate the events.
-
-

In the following example, the billing date is the first day of the month:

On February 15, the free minutes resource is adjusted from 100 to 500 and the adjustment is backdated to be effective January 15. The **backdate_window** is 24 hours and **num_billing_cycles** is 1 cycle. When **pin_rerate** is run on February 20 with the **-rerate** parameter, BRM automatically rerates all the relevant events from January 15 to February 20 and applies the balance changes to the current billing cycle. It creates adjustment events for the events that occurred from January 15 to February 1 and shadow events for the events that occurred from February 1 to February 20. Billing events that previously occurred on February 1, such as billing-time discounts, rollovers, and folds, are recalculated based on the new adjustment and are reapplied.

About Backdated ERA Modifications

BRM automatically rerates backdated ERA modifications when the validity start time or end time is backdated and automatic rerating is enabled.

For example, a subscriber changes their service-level agreement from Silver to Gold on July 12. The CSR backdates the change to July 1 to let the subscriber apply the Gold-level benefits to all usage for July. The next time you run `pin_rerate`, BRM rerates all the relevant events for the account that occurred between July 1 and the current time.

Configuring Automatic Rerating of Backdated Events

To configure automatic rerating of backdated events, you perform the following tasks:

- [Setting Thresholds That Trigger Automatic Rerating](#)
- [Configuring Event Notification for Rerating Backdated Events](#)

Setting Thresholds That Trigger Automatic Rerating

BRM automatically rerates certain backdated events based on the default CM configuration settings. See "[About Automatic Rerating of Backdated Events](#)".

To change the default settings:

1. Open the CM configuration file (`BRM_Home/sys/cm/pin.conf`) in a text editor.
2. To turn automatic rerating on or off, set the `backdate_trigger_auto_rerate` entry: `1` = enabled; `0` = disabled.


```
- fm_subs backdate_trigger_auto_rerate 1
```
3. To specify the minimum time difference necessary to trigger automatic rerating, set the `backdate_window` entry. This is the amount of time in seconds between the current time and the backdated time. The default is 3600 seconds.


```
- fm_subs backdate_window 3600
```
4. To specify the maximum number of billing cycles allowed between the current time and the backdated event date for triggering automatic rerating, set the `num_billing_cycles` entry. The default is 1 billing cycle.


```
- fm_subs num_billing_cycles 1
```
5. Save and close the file.
6. Stop and restart the CM.

Configuring Event Notification for Rerating Backdated Events

When a backdated event occurs, BRM rerating uses event notification to trigger automatic rerating of the event.

Although any subclass of the `/event` class can be used to trigger event notification, BRM rerating generates the nonpersistent `/event/notification/auto_rerate` event specifically to use for event notification.

By default, when this event occurs, BRM creates a `rerate` job.

Before you can use BRM rerating, you must configure the event notification feature as follows:

1. If your system has multiple configuration files for event notification, merge them.
2. Ensure that the merged file includes the following information from the `BRM_Home/sys/data/config/pin_notify` file:

```
# Rerating related event notification
```

```
3787 0 /event/notification/auto_rerate
```

3. (Optional) If necessary to accommodate your business needs, add, modify, or delete entries in your final event notification list.
4. (Optional) If necessary to accommodate your business needs, create custom code for event notification to trigger.
5. Load your final event notification list into the BRM database.

For more information, see "Using Event Notification" in *BRM Developer's Guide*.

Backdating beyond Configured Billing Cycles without Automatic Rerating Request

You can backdate an event beyond the configured number of billing cycles without requesting to automatically rerate such an event.

To do so:

1. Ensure that automatic rerating is enabled for backdated events. If necessary, enable it by setting the **backdate_trigger_auto_rerate** entry in CM configuration file to **1**. For more information on **backdate_trigger_auto_rerate**, see the description of pricing and rating **pin.conf** entries in *BRM System Administrator's Guide*.
2. Enable the **AllowBackdateNoRerate** business parameter in the **/config/business_params** object by using the **pin_bus_params** utility. For more information, see **pin_bus_params** in *System Administrator's Guide*.
3. After you enable the **AllowBackdateNoRerate** business parameter, you must manually rerate any events backdated *beyond* the number of billing cycles specified in the **num_billing_cycles** entry.

For information on manually rerating events, see "[Using the pin_rerate Utility](#)".

For information on **num_billing_cycles**, see the description of Billing **pin.conf** entries in *BRM System Administrator's Guide*.

Enabling the AllowBackdateNoRerate Business Parameter

To set the **AllowBackdateNoRerate** business parameter to **enabled**:

1. Go to *BRM_Home/sys/data/config* directory.
2. Create an editable XML file of the subscription instance from the **/config/business_params** object by using the following command:

```
pin_bus_params -r -c "Subscription" bus_params_subscription.xml
```

BRM places the XML file named **bus_params_subscription.xml.out** in your working directory. To place this file in a different directory, specify the full path as part of the file name.

3. Locate the **AllowBackdateNoRerate** entry in the **bus_params_subscription.xml.out** file.
4. Set the value of **AllowBackdateNoRerate** to **enabled**, if necessary.

```
<AllowBackdateNoRerate>enabled</AllowBackdateNoRerate>
```

Caution: BRM uses the XML in this file to overwrite the existing subscription 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 subscription configuration.

5. Save this updated file as `bus_params_subscription.xml`.
6. Load the modified XML file containing the business parameters for billing into the appropriate `/config/business_params` object in the BRM database.

pin_bus_params bus_params_subscription.xml

You should execute this command from the `BRM_Home/sys/data/config` directory, which includes support files used by the utility. To execute it from a different directory, see the description of `pin_bus_params` in *BRM System Administrator's Guide*.

7. Read the object with the `testnap` utility or Object Browser to verify that all fields are correct. BRM stores one of the following values for `AllowBackdateNoRerate`:
 - `0` to indicate **disabled**.
 - `1` to indicate **enabled**.

For more information on reading objects by using the Object Browser, see *BRM Managing Customers*. For instructions on using the `testnap` utility, see *BRM Developer's Guide*.

8. Stop and restart Connection Manager. See "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.
9. (Multischema systems only) Run the `pin_multidb` script with the `-R CONFIG` parameter. For more information, see "pin_multidb" in *BRM System Administrator's Guide*.

About Automatic Rerating of Out-of-Order Events

Events are processed by Pipeline Manager in the order that call details records (CDRs) are received. If the CDRs are sent out of order to Pipeline Manager, the events are processed out of order as well. Usually, this is not a problem; however, correct rating sometimes depends on rating events in chronological order (for example, when usage counters are used).

You can configure BRM to detect events that must be rated in chronological order and rerate them. To use out-of-order rerating, you define the criteria for when an out-of-order event must be rerated. When an event is rated, the `FCT_EventOrder` module uses the criteria and the event timestamps to determine if the event needs to be rerated.

About Detecting Out-of-Order Events

To enable out-of-order rerating, you configure criteria BRM uses to detect when events qualify for out-of-order rerating. Events must be rated in chronological order only when resources from the same balance group are affected, so the balance group is assumed as part of the detection criteria. If an account owns more than one service instance, each using a different balance group, out-of-order detection is applied to only the balance group associated with the event (the criteria name and balance group combination that is stored in the `/profile/event_ordering` object).

Pipeline Manager loads your detection criteria configuration at startup as follows:

- The DAT_PortalConfig module retrieves data from the `/config/event_order_criteria` object and loads it into its memory. This data specifies the criteria for determining if an event qualifies for out-of-order detection. The criteria is based on:
 - Service types
 - Event types
 - Products
 - Discounts
 - A combination of service types, event types, products, and discounts

For more information, see ["About Out-of-Order Rerating Criteria"](#).

Note: This feature supports branding. You can have a different configuration for each brand.

- The DAT_AccountBatch module retrieves data from the `/profile/event_ordering` profile object and loads it into its memory. This data includes the timestamp for the last time an event was processed for a particular billing cycle with the criteria specified in the `/config/event_order_criteria` object and the balance group determined when an EDR is found to be out of order.

Note: If an out-of-order event comes in from a previous billing cycle after billing has run for that cycle (for example, when delayed billing is configured), the event is handled by default as if it is part of the current billing cycle.

How BRM Rerates Out-of-Order Events

The overall process for rerating out-of-order events is as follows:

1. Pipeline Manager loads your detection criteria configuration at startup. See ["About Detecting Out-of-Order Events"](#).
2. An event is rated in the pipeline and is rated by the rating and discounting modules.
3. The FCT_EventOrder module gets the out-of-order detection criteria from the DAT_PortalConfig module to determine if the event needs to be rerated.

In addition, FCT_EventOrder gets data from the DAT_AccountBatch module that specifies the latest event processed time for each appropriate criterion and balance group combination for an active billing cycle.
4. FCT_EventOrder determines whether the event needs to be rerated:
 - If the event is out of order, the module proceeds to the next step.
 - If the event is not out of order, FCT_EventOrder triggers an update to the last event processed time in the DAT_AccountBatch memory. A record (record type = 850) is added to the pipeline output to update the `/profile/event_ordering` object with the new event's start time.
5. The event data record (EDR) is kept in the FCT_EventOrder module's shared memory until after the transaction commits, when it is then sent to a rerate-request

file. When Batch Controller detects this file, it starts the OODHandler batch handler to process it. The rerate-request file contains multiple accounts for which out-of-order events were detected.

Note:

- You can configure FCT_EventOrder to write the out-of-order EDR data to separate rerate-request files after the transaction commits; batching the requests in this way helps performance in Pipeline Manager. See "[Batching Out-of-Order Rerate Jobs](#)".
 - You can configure FCT_EventOrder to write a specific number of accounts to each rerate-request file. See "[Configuring the Number of Accounts in an Out-of-Order Rerate Job](#)".
-
-

Rerate-request file names use the following format:

outputPrefix_pipelineName_transactionID_sequenceNumber.xml

where:

outputPrefix is the prefix specified by the **OutputPrefix** entry in the FCT_EventOrder module registry. The default is **ood**.

pipelineName is the name of the pipeline; in the following example, the name is ALL_RATE.

transactionID is the transaction ID.

sequenceNumber is the sequence number of the job.

For example:

ood_ALL_RATE_14_0.xml

6. The OODHandler batch handler processes the out-of-order rerate-request file, moves it to the *BRM_Home/apps/pin_ood_handler/process* directory, and calls the "[pin_load_rerate_jobs](#)" utility.
7. The **pin_load_rerate_jobs** utility creates an input flist with the following rerate output file data:
 - Account
 - Balance Group
 - CriteriaName
 - Rerate Start Time
 - Service

It then calls the PCM_OP_ACT_USAGE opcode in calc-only mode, which generates a notification event (*/event/notification/activity/out_of_order*) that triggers the PCM_OP_ACT_HANDLE_OOD_EVENT opcode. PCM_OP_ACT_USAGE passes the data to PCM_OP_ACT_HANDLE_OOD_EVENT.

8. PCM_OP_ACT_HANDLE_OOD_EVENT calls the PCM_OP_RERATE_INSERT_RERATE_REQUEST opcode.
9. PCM_OP_RERATE_INSERT_RERATE_REQUEST checks for duplicate rerate jobs in the rerate-request file and calls other opcodes to create a rerate job out of the file.

10. The rated event is loaded into the BRM database by Rated Event (RE) Loader. If the event qualifies for out-of-order detection (and the event is in order), RE Loader updates the account's `/profile/event_ordering` object.

Note: The `/profile/event_ordering` object stores data for the current and next billing cycles and for closed billing cycles. To clean up data for closed billing cycles, see "[Purging Event Ordering Profile Data for Closed Billing Cycles](#)".

About Out-of-Order Rerating Criteria

When the FCT_EventOrder module checks for out-of-order events, it uses the following criteria:

- The timestamp of the event. If the event is later than the latest event that is under consideration, the event is not out of order.
- The out-of-order detection criteria in which the event is defined in the `/config/event_order_criteria` object. If the event is not defined in any criteria in the `/config/event_order_criteria` object, it is assumed the event does not need to be rated in chronological order.
- The last event processed time for each balance group and the name of the trigger-dependant rerating criteria for that event for a billing cycle in the account's `/profile/event_ordering` object. If the event has a later timestamp than the latest event processed time for an event that uses the same balance group and criteria name, the event is not out of order.

Define the criteria for the events you must rate in chronological order in the `/config/event_order_criteria` object. The data stored in the `/config/event_order_criteria` object consists of two parts:

- The name of the criterion.
- One or more parameters, such as a service or event type or a product name or discount name.

Your configuration can use as many parameters as you require. [Table 29–2](#) provides examples of out-of-order criteria and what they mean for out-of-order detection:

Table 29–2 Criteria for Out-of-Order Detection

Criteria	Out-of-Order Detection
Criteria name: GSM_TEL_SERVICE Parameter: <code>/service/gsm/telephony</code>	All events for the GSM telephony service should be considered for out-of-order rerating.
Criteria name: GSM_EVENTS Parameter: <code>/event/delayed/session/telco/gsm/telephony</code> Parameter: <code>/event/delayed/session/telco/gsm/sms</code> Parameter: <code>/event/delayed/session/telco/gsm/fax</code>	All GSM events should be considered for out-of-order rerating.
Criteria name: GSM_Product Parameter: GSM Voice Product	All events for a specific product should be considered for out-of-order rerating. All EDRs that are rated by the GSM Voice Product will be considered for out-of-order rerating.

For detailed information on defining your criteria, see ["Defining Out-of-Order Criteria"](#).

Setting Up Out-of-Order Rerating

To set up out-of-order rerating, perform the tasks in the following sections:

- [Defining Out-of-Order Criteria](#)
- [Loading Out-of-Order Criteria](#)

Important: Accounts created *after* you load the `/config/event_order_criteria` object in the BRM database qualify for out-of-order detection.

- [Configuring Out-of-Order Detection in a Pipeline.](#)
- [Configuring Event Notification for Out-of-Order Rerating](#)
- (Optional) ["Specifying a Reason Code for Rerating Out-of-Order Events"](#)
- [Configuring Batch Controller for Rerating Out-of-Order Events](#)
- [Configuring the OODHandler Batch Handler for Rerating Out-of-Order Events](#)
- (Optional) ["Purging Event Ordering Profile Data for Closed Billing Cycles"](#)

Defining Out-of-Order Criteria

When you define out-of-order criteria, you specify parameters for events, services, discount names, or product names that qualify for out-of-order detection. The parameters you specify are the criteria BRM uses to ensure that events are rated in chronological order when you want them to be. For more information, see ["About Out-of-Order Rerating Criteria"](#).

Important: You should know how your price plans are structured to track resources, such as resources for a specific service or resources for a specific type of usage. Tracking resources can involve using service-level balance groups and a criterion is set up for events that use the same balance group.

For more information, see ["Tracking Resources by Service"](#) and ["About Tracking Resources in Account Sub-Balances"](#).

You typically list parameters that use the same balance group under the same criterion name. Services that use different balance groups are listed under separate criterion names. The `/profile/event_ordering` object stores the latest event processed time for each appropriate criteria and balance group combination for each active billing cycle. For more information, see ["Defining Criteria under Separate Criteria Names"](#).

You define out-of-order criteria in the `pin_config_ood_criteria.xml` file in `BRM_Home/sys/data/config`. For a sample of this file, see ["Sample Out-of-Order Criteria File"](#).

You can define criteria by using one or more parameters.

- The simplest criteria uses a single parameter.

For example:

```
- <OodCriteriaElement>
```

```
<CriteriaName>GSM_TEL_SERVICE</CriteriaName>
- <ParameterList>
<Parameter Type="Service">/service/gsm/telephony</Parameter>
</ParameterList>
</OodCriteriaElement>
```

- You can use two parameters in the following combinations:
 - Service Type, Event Type
 - Service Type, Product
 - Service Type, Discount
 - Event Type, Product
 - Event Type, Discount

For example:

```
- <OodCriteriaElement>
<CriteriaName>GSM_TEL_SERVICE</CriteriaName>
- <ParameterList>
<Parameter Type="Service">/service/gsm/telephony</Parameter>
<Parameter Type="Event">/event/delayed/session/gsm/telephony</Parameter>
</ParameterList>
</OodCriteriaElement>
```

- You can use multiple parameters to combine service type, event type, product, and discount.

For example:

```
- <OodCriteriaElement>
<CriteriaName>GSM</CriteriaName>
- <ParameterList>
<Parameter Type="Service">/service/telco/gsm/telephony</Parameter>
<Parameter Type="Service">/service/telco/gsm/sms</Parameter>
<Parameter Type="Event">/event/delayed/session/telco/gsm/telephony</Parameter>
<Parameter Type="Event">/event/delayed/session/telco/gsm/sms</Parameter>
<Parameter Type="Product">GSM Voice Product</Parameter>
<Parameter Type="Product">GSM Voice Discount</Parameter>
</ParameterList>
</OodCriteriaElement>
```

If you use a combination of event, service, and product or discount, the EDR parameters must match all of the following:

- One service type (for example, service type 1 or service type 2)
- One event type (for example, event type 1 or event type 2)
- One product or discount (for example, product 1 or product 2 or discount 1 or discount 2)

For example, in the following detection criterion named GSM_Multi:

```
- <OodCriteriaElement>
<CriteriaName>GSM_Multi</CriteriaName>
- <ParameterList>
<Parameter Type="Service">/service/telco/gsm/telephony</Parameter>
<Parameter Type="Service">/service/telco/gsm/sms</Parameter>
<Parameter Type="Event">/event/delayed/session/telco/gsm/telephony</Parameter>
<Parameter Type="Event">/event/delayed/session/telco/gsm/sms</Parameter>
<Parameter Type="Product">GSM Voice Product Telephony</Parameter>
```

```
<Parameter Type="Discount">GSM Voice Product SMS</Parameter>
</ParameterList>
</OodCriteriaElement>
```

The EDR must match the following parameters:

- Service type **/service/telco/gsm/telephony**
- Event type **/event/delayed/session/telco/gsm/telephony**
- Product GSM Voice Product Telephony

Or the following parameters:

- Service type **/service/telco/gsm/sms**
- Event type **/event/delayed/session/telco/gsm/sms**
- Product GSM Voice Product SMS

That means that when an EDR arrives for that particular service rated by that particular product when that particular event occurs, BRM considers it for out-of-order rerating.

Defining Criteria under Separate Criteria Names

List services that use the same balance group under the same criteria name. BRM does not differentiate between the events of these services when obtaining the latest event processed time in the account's profile object because they consume resources in the same way for rating.

The following out-of-order criteria would apply to two GSM services that share the same balance group:

```
- <OodCriteriaElement>
<CriteriaName>GSM</CriteriaName>
- <ParameterList>
<Parameter Type="Service">/service/telco/gsm/telephony</Parameter>
<Parameter Type="Service">/service/telco/gsm/sms</Parameter>
</ParameterList>
</OodCriteriaElement>
```

If the latest event processed time is later than an EDR event timestamp, the event needs to be rerated regardless of the service to which it belongs; for example, for the preceding GSM criteria, if the latest event processed time is 2 p.m. for an SMS event and the EDR is a telephony event with a 1 p.m. timestamp, the event is out of order and needs to be rerated.

If an account can own more than one instance of a service and each instance uses a different balance group, out-of-order rerating is applied to the balance group of the service instance to which the event belongs. Accounts can have multiple balance groups if you offer plans in which service-level balance groups are used (see ["Tracking Resources by Service"](#) for more information). When events for these services must be rated in chronological order, you must evaluate the out-of-order criteria for these services separately from each other.

You list services that use different balance groups under separate criteria names. The following out-of-order criteria would apply to two GSM services that use different balance groups:

```
- <OodCriteriaElement>
<CriteriaName>GSM_TEL</CriteriaName>
- <ParameterList>
<Parameter Type="Service">/service/telco/gsm/telephony</Parameter>
```

```

</ParameterList>
</OodCriteriaElement>
- <OodCriteriaElement>
<CriteriaName>GSM_SMS</CriteriaName>
- <ParameterList>
<Parameter Type="Service">/service/telco/gsm/sms</Parameter>
</ParameterList>
</OodCriteriaElement>

```

Because the latest event processed times for all services used by an account are tracked in one profile object, BRM uses the unique criteria name to differentiate between the service-specific latest event processed times. Thus, in the preceding criteria, BRM uses the criteria name GSM_TEL in the account's profile object to track the latest event processed time for the telephony events and uses the criteria name GSM_SMS to track the latest event processed time for the SMS events.

There may be times when you must define parameters under separate criterion names when they share the same balance group but you are tracking different resources. For example, if a GSM telephony service and a GSM discount share a balance group and are rated by the same event, you would define a criterion for the discount if you are tracking the last time resources were consumed with that particular discount.

Avoiding Overlapping Criteria

If you configure criteria already contained in another criteria, the ["load_pin_config_ood_criteria"](#) utility reports an error, and nothing is loaded into the database. In the following example, the GSM_TEL_1 criteria is a superset of the GSM_TEL_2 criteria, so GSM_TEL_2 is considered an overlapping criteria:

```

- <OodCriteriaElement>
<CriteriaName>GSM_TEL_1</CriteriaName>
- <ParameterList>
<Parameter Type="Service">/service/telco/gsm/telephony</Parameter>
</ParameterList>
</OodCriteriaElement>
- <OodCriteriaElement>
<CriteriaName>GSM_TEL_2</CriteriaName>
- <ParameterList>
<Parameter Type="Service">/service/telco/gsm/telephony</Parameter>
<Parameter Type="Event">/event/delayed/session/telco/gsm/telephony</Parameter>
</ParameterList>
</OodCriteriaElement>

```

In the following example, the GSM_Dual_A criteria is a superset of the GSM_Multi_B criteria, so GSM_Multi_B is an overlapping criteria:

```

- <OodCriteriaElement>
<CriteriaName>GSM_Dual_A</CriteriaName>
- <ParameterList>
<Parameter Type="Service">/service/telco/gsm/telephony</Parameter>
<Parameter Type="Service">/service/telco/gsm/sms</Parameter>
<Parameter Type="Product">GSM Voice Product Telephony</Parameter>
<Parameter Type="Discount">GSM Voice Discount SMS</Parameter>
</ParameterList>
</OodCriteriaElement>
- <OodCriteriaElement>
<CriteriaName>GSM_Multi_B</CriteriaName>
- <ParameterList>
<Parameter Type="Service">/service/telco/gsm/telephony</Parameter>
<Parameter Type="Service">/service/telco/gsm/sms</Parameter>
<Parameter Type="Event">/event/delayed/session/telco/gsm/telephony</Parameter>

```

```

<Parameter Type="Event">/event/delayed/session/telco/gsm/sms</Parameter>
<Parameter Type="Product">GSM Voice Product Telephony</Parameter>
<Parameter Type="Discount">GSM Voice Discount SMS</Parameter>
</ParameterList>
</OodCriteriaElement>

```

You would choose one of the preceding criterion based on your goals. You would never use both of them. For example:

- You would configure the GSM_Dual_A criterion if you wanted BRM to consider EDRs for out-of-order rerating when they matched the following:
 - Any event type for the GSM telephony service that is also rated by the GSM Voice Product Telephony product.

or

 - Any event type for the GSM SMS service that is also rated by the GSM Voice Discount SMS discount.
- You would configure the GSM_Multi_B criterion if you wanted BRM to consider EDRs for out-of-order rerating when they matched the following:
 - The specific event type `/event/delayed/session/telco/gsm/telephony` for the GSM telephony service that is also rated by the GSM Voice Product Telephony product.

or

 - The specific event type `/event/delayed/session/telco/gsm/sms` for the GSM SMS service that is also rated by the GSM Voice Discount SMS discount.

Because GSM_Dual_A is configured to consider all event types including the specific event types configured in GSM_Multi_B, the out-of-order detection configured in GSM_Multi_B is already handled by the GSM_Dual_A criterion. If you only want those specific event types to be checked for out-of-order detection, you would use GSM_Multi_B rather than GSM_Dual_A.

Sample Out-of-Order Criteria File

The following is a sample configuration of the `pin_config_ood_criteria.xml` file.

```

<?xml version="1.0" encoding="UTF-8" ?>
- <OodCriteriaConfiguration xmlns="http://www.portal.com/schemas/BusinessConfig"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.portal.com/schemas/BusinessConfig pin_config_ood_
criteria.xsd">

- <!-- This file is a sample file. -->
- <!-- This file needs to be modified based on the out-of-order configuration
required -->
- <!-- for an installation. THIS FILE SHOULD NOT BE LOADED WITH THE DATA SUPPLIED
HERE. -->

- <OodCriteriaElement>
<CriteriaName>TEL_SERVICE</CriteriaName>
- <ParameterList>
<Parameter Type="Service">/service/telco/gsm/telephony</Parameter>
<Parameter Type="Event">/event/delayed/session/telco/gsm/telephony</Parameter>
</ParameterList>
</OodCriteriaElement>

- <OodCriteriaElement>

```

```

<CriteriaName>GSM_SERVICE</CriteriaName>
- <ParameterList>
<Parameter Type="Service">/service/telco/gsm/sms</Parameter>
<Parameter Type="Event">/event/delayed/session/telco/gsm/sms</Parameter>
</ParameterList>
</OodCriteriaElement>

- <OodCriteriaElement>
<CriteriaName>A_MULTI_TEL_SERVICE</CriteriaName>
- <ParameterList>
<Parameter Type="Service">/service/telco/gsm</Parameter>
<Parameter Type="Product">Standard GSM Telephony</Parameter>
</ParameterList>
</OodCriteriaElement>

- <OodCriteriaElement>
<CriteriaName>A_MULTI_TEL_SERVICE1</CriteriaName>
- <ParameterList>
<Parameter Type="Service">/service/telco/gsm</Parameter>
<Parameter Type="Discount">Standard GSM Discount</Parameter>
</ParameterList>
</OodCriteriaElement>

- <OodCriteriaElement>
<CriteriaName>TEL_SERVICE_EVENT</CriteriaName>
- <ParameterList>
<Parameter Type="Event">/event/delayed/session/telco/gsm</Parameter>
</ParameterList>
</OodCriteriaElement>

</OodCriteriaConfiguration>

```

Loading Out-of-Order Criteria

To configure out-of-order criteria, you edit the `pin_config_ood_criteria.xml` file and load the contents of the file into the BRM database by using the `load_pin_config_ood_criteria` utility. The data is stored in the `/config/event_order_criteria` object.

Important: To connect to the BRM database, the `load_pin_config_ood_criteria` utility needs a configuration (`pin.conf`) file in the directory from which you run the utility.

Caution: The `load_pin_config_ood_criteria` utility overwrites existing out-of-order criteria. If you are updating out-of-order criteria, you cannot load new out-of-order criteria only. You must load complete sets of out-of-order criteria each time you run the `load_pin_config_ood_criteria` utility.

Note: You can run this utility to configure out-of-order criteria for different brands.

1. Edit the `pin_config_ood_criteria.xml` file in `BRM_Home/sys/data/config`. For a sample of this file, see "[Sample Out-of-Order Criteria File](#)".

2. Save and close the file.
3. Use the following command to run the `load_pin_config_ood_criteria` utility:

```
load_pin_config_ood_criteria pin_config_ood_criteria.xml
```

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_config_ood_criteria BRM_Home/sys/data/config/pin_config_ood_criteria.xml
```

Tip: If you copy the `pin_config_ood_criteria.xml` file to the directory from which you run the `load_pin_config_ood_criteria` utility, you do not have to specify the path or file name. By default, the file is named `pin_config_ood_criteria.xml`. You can change this name.

For more information, see "`load_pin_config_ood_criteria`".

4. If Pipeline Manager is running, send a CBPREload semaphore to reload data. The **Reload** semaphore is used by the DAT_PortalConfig data module. If Pipeline Manager is not running, it will load the data the next time it is started.

To verify that the out-of-order criteria were loaded, you can display the `/config/event_order_criteria` object by using Object Browser, or use the `robj` command with the `testnap` utility.

Configuring Out-of-Order Detection in a Pipeline

To configure out-of-order detection in a pipeline, you do the following:

1. Configure the DAT_PortalConfig module.
2. Configure the FCT_EventOrder module.

Important: FCT_EventOrder should be located in the pipeline *after* the rating (FCT_MainRating) and discounting (FCT_Discount) modules and *before* the rejection (FCT_Reject) module.

When configuring FCT_EventOrder, specify the following:

- The amount of data to include in each rerate request file. See "[Batching Out-of-Order Rerate Jobs](#)".
- The number of accounts assigned to each rerate job. See "[Configuring the Number of Accounts in an Out-of-Order Rerate Job](#)".

Batching Out-of-Order Rerate Jobs

You can configure the FCT_EventOrder module to batch the out-of-order EDR data in its transactional memory and write it to separate rerate-request files after the transaction commits. Batching rerate jobs in this way improves performance during rerating. To control how many records FCT_EventOrder batches into separate rerate request files, you specify an amount of time in minutes using the `RerateDelayTolerance` registry entry.

When the transaction commits, FCT_EventOrder sorts the rerate-request data in its transactional memory based on the EDR start time. The rerate start time of the first record in the rerate-request file is the rerate start time for the rerate job. As FCT_

EventOrder writes the events into the rerate-request file, it compares the start time of the first record with the start time of each event being added. If the time difference exceeds the value of the **RerateDelayTolerance** entry, that event becomes the last record in the rerate-request file, and a new rerate-request file is created.

Example 1

If the **RerateDelayTolerance** entry is set to 180 minutes, and the start time of the first event in the rerate-request file is 3:15 p.m, events are added until the start time of an event is greater than 6:15 p.m.

Example 2

If the **RerateDelayTolerance** entry is set to 30 minutes, and the start time of the first event in the rerate-request file is 11:00 a.m. and four subsequent events have start times of 11:10 a.m., 12:05 p.m., 12:15 p.m., and 12:33 p.m, FCT_EventOrder creates two rerate-request files, resulting in two rerate jobs (**/job_batch/rerate** objects) as follows:

- Rerate job 1 contains accounts associated with EDRs that have these start times:
 - 11:00 a.m.
 - 11:10 a.m.
- Rerate job 2 contains accounts associated with EDRs that have these start times:
 - 12:05 p.m.
 - 12:15 p.m.
 - 12:33 p.m.

In the preceding example, if for one account the first EDR arrives at 11:10 a.m. and the last EDR arrives at 11:45 a.m., the account is included in Rerate job 1.

Configuring the Number of Accounts in an Out-of-Order Rerate Job

To improve batch rerating throughput, you can specify the number of accounts FCT_EventOrder assigns to each rerate job by using the **NumberOfAccountLimit** registry entry. FCT_EventOrder writes out the number of accounts specified by this entry to the rerate-request file, which, in turn, becomes the rerate job after detection of duplicate rerate requests is complete.

The default for the **NumberOfAccountLimit** registry entry is 1000.

Important: The **NumberOfAccountLimit** registry entry of FCT_EventOrder is similar to the **per_job** configuration entry of the **pin_rerate** utility. When you configure these entries, BRM recommends you use the same value.

For more information, see ["Configuring the Number of Accounts Per Job and Number of Jobs per Transaction"](#).

Configuring Event Notification for Out-of-Order Rerating

To create rerate jobs automatically when certain events are not in chronological order, you must configure event notification for out-of-order rerating. BRM uses the **/event/notification/activity/out_of_order** event to trigger automatic rerating when an out-of-order event occurs.

To configure event notification for out-of-order rerating, do the following:

1. If your system has multiple configuration files for event notification, merge them.

All of the event notification configuration files available in your system are in the *BRM_Home/sys/data/config* directory.

Depending on which BRM features you use, your system may contain one or more configuration files for event notification; for example, a wireless system may use **pin_notify_ifw_sync** (supports Account Synchronization and standard recycling) or **pin_notify_telco** (supports GSM Manager). If your system contains more than one of these files, you must merge their contents into a single file.

2. Add the following entry to the merged file:

```
# Event notification for out-of-order rerating
177 0 /event/notification/activity/out_of_order
```

This configures the **/event/notification/activity/out_of_order** event to trigger rerating and to call the **PCM_OP_ACT_HANDLE_OOD_EVENT** opcode (opcode ID number 177) to select which events to rerate.

PCM_OP_ACT_HANDLE_OOD_EVENT calls the **PCM_OP_RERATE_INSERT_RERATE_REQUEST** opcode to insert a rerate job for the out-of-order events.

Note: You can configure this notification event to call a custom opcode. The custom opcode can analyze the event, determine if rerating is required, and then call **PCM_OP_RERATE_INSERT_RERATE_REQUEST** to create the rerate job with a rerate reason. For more information, see ["Setting Up Trigger-Dependent Rerating"](#).

3. (Optional) If necessary to accommodate your business needs, add, modify, or delete entries in your final event notification list.
4. (Optional) If necessary to accommodate your business needs, create custom code for event notification to trigger.
5. Load your final event notification list into the BRM database by running the **load_pin_notify** utility to load the contents of the file into the **/config/notify** object.

For instructions on using this utility, see "Using Event Notification" in *BRM Developer's Guide*.

Specifying a Reason Code for Rerating Out-of-Order Events

Rerate jobs can be assigned a rerate reason code. When you run the **pin_rerate** utility, you can rerate all accounts for a rerate job with a specific reason code. By default, rerate jobs for out-of-order events have the reason code **1**. You can change this to any number you want in the CM **pin.conf** file by using the following entry:

```
fm_act ood_rerate_job_reason_code rerate_reason_code
```

Important: Rerate reason codes can also be assigned by other BRM automatic rerating features and when you run **pin_rerate**. To rerate accounts according to type of rerate reason code, ensure that your rerate reason codes are unique.

For more information about processing rerate jobs according to a rerate reason code, see ["About Processing Rerate Jobs Created by Automatic Rerating"](#).

Configuring Batch Controller for Rerating Out-of-Order Events

The OODHandler batch handler is run automatically by Batch Controller. You must configure Batch Controller and the OODHandler batch handler.

To configure Batch Controller to rerate out-of-order events:

1. Open the Batch Controller `Infranet.properties` file in `BRM_Home/apps/batch_controller`.
2. Add the following entries listed in [Table 29–3](#) for the **OODHandler** batch handler:

Table 29–3 Entries for OODHandler Batch Handler

Entry	Description
<code>batch.random.events</code>	Specify your event identifier in the event identifier list. For example: <pre>batch.random.events TEL, SMS, FAX, OODHANDLER</pre> where OODHANDLER is your event identifier.
<code>event_identifier.name</code>	Specify a description for the event identifier. For example: <pre>OODHANDLER.name OODHANDLER usage</pre>
<code>event_identifier.file.location</code>	Specify the path to the directory where Pipeline Manager puts the rerate request files. The location of this directory is configured in the OutputDirectory entry in the <code>FCT_EventOrder</code> module wireless registry file (<code>Pipeline_home/conf/wireless.reg</code>). Important: You must create the directory. It is not created by BRM installation scripts. For example: <pre>OODHANDLER.file.location Pipeline_ home/data/out/ood</pre>
<code>event_identifier.file.pattern</code>	Specify the rerate job output file name. When Batch Controller detects a file with this name, it starts the batch handler. Tip: You can use an asterisk (*) to represent zero or more characters in the file name. No other wildcards are supported. For example: <pre>OODHANDLER.file.pattern *.xml</pre>
<code>event_identifier.handlers</code>	Specify the batch handler identifier. For example: <pre>OODHANDLER.handlers OodHandler</pre>
<code>handler_identifier.name</code>	Specify a description for the batch handler identifier. For example: <pre>OodHandler.name OODHandler</pre>
<code>handler_identifier.max.at.lowload.time</code> <code>handler_identifier.max.at.highload.time</code>	Specify the number of batch handler instances that can run concurrently during periods of low load and high load usage. Typical default settings are 6 at low load and 3 at high load. For example: <pre>OodHandler.max.at.lowload.time 6 OodHandler.max.at.highload.time 3</pre>

Table 29–3 (Cont.) Entries for OODHandler Batch Handler

Entry	Description
<code>handler_identifier.start.string</code>	<p>Specify the command that starts the OODHandler batch handler. For example, the default is:</p> <pre>OodHandler.start.string BRM_ Home/apps/pin_ood_handler/OODHandler.pl.</pre> <p>Important: Copy the <code>BRM_Home/bin/OODHandler</code> to <code>BRM_Home/apps/pin_ood_handler/OODHandler.pl</code>.</p>

3. Save and close the file.

Configuring the OODHandler Batch Handler for Rerating Out-of-Order Events

The OODHandler batch handler retrieves the rerate-request file from the Pipeline Manager output and runs the `pin_load_rerate_jobs` utility to create rerate jobs. After the rerate jobs are created, the batch handler moves the rerate-request file to a different directory.

Important: If you use the `ConfigurableValidityHandler` batch handler for loading first-usage validity data, do not use the `OODHandler_config.values` file as instructed below. Instead, you must configure the OODHandler batch handler in the `ConfigurableValidityHandler` configuration file (`BRM_Home/apps/pin_rel/ConfigurableValidityHandler_config.values`). `ConfigurableValidityHandler` runs both the `pin_load_rerate_jobs` utility and the utility for loading validity data.

To configure the OODHandler batch handler:

1. Open the `OodHandler_config.values` configuration file in `BRM_Home/apps/pin_ood_handler`.
2. Edit the following entries shown in [Table 29–4](#). For information about other entries, see the `OodHandler_config.values` file.

Table 29–4 Entries to Configure OodHandler Batch Handler

Entry	Description
<code>\$FILETYPE</code>	<p>Specify the file name pattern of the rerate-request file. For example:</p> <pre>\$FILETYPE = "*.xml.bc";</pre> <p>Note: The asterisk (*) represents zero or more characters in the file name. No other wildcards are supported.</p> <p>Batch Controller runs the OODHandler batch handler for each file with a name that matches this pattern.</p> <p>Important: The file name pattern must end with the <code>.bc</code> extension. Batch Controller automatically appends <code>.bc</code> to each file name before it runs a batch handler.</p>

Table 29–4 (Cont.) Entries to Configure OodHandler Batch Handler

Entry	Description
\$HANDLER_DIR	Specify the path to the directory containing the OODHandler batch handler configuration files, log files, and other processing files and directories. The default is <i>BRM_Home/apps/pin_ood_handler</i> .
\$pinLoadRerateJobsDir	Specify the path to the directory where the pin_load_rerate_jobs utility processes the files (this contains the pin_rerate_job_info.xsd file). The default is <i>BRM_Home/apps/pin_ood_handler/process</i> .
\$pinLOADRRERATEJOBS	Specify the application run by the OODHandler batch handler to process the files. For example: <code>\$pinLOADRRERATEJOBS = "pin_load_rerate_jobs";</code>
\$STAGING	Specify the full path to the OODHandler batch handler rerate-request file location. For example: <code>\$STAGING = "\$Pipeline_home/data/out/ood";</code> This is the same directory where Pipeline Manager puts the rerate-request files. When Batch Controller detects the rerate-request files in this location, it calls the OODHandler batch handler. Note: The location of this directory is also configured in the OutputDirectory entry in the FCT_EventOrder module wireless registry file (<i>Pipeline_home/conf/wireless.reg</i>). You must create the directory. It is not created by BRM installation scripts.
\$PROCESSING	Specify the full path to the directory from which the rerate job files are processed by the OODHandler batch handler. The default is \$pinLoadRerateJobsDir . The OODHandler batch handler takes the files from the \$STAGING directory and places them here.
\$LOGFILE	Specify the full path to the OODHandler batch handler log file. For example: <code>\$LOGFILE = "\$HANDLER_DIR/OOD_Handler.log";</code>

3. Save and close the file.

Purging Event Ordering Profile Data for Closed Billing Cycles

If out-of-order detection is configured in the system, one instance of the **/profile/event_ordering** object is created for each account during customer creation. The profile exists for the lifetime of the account, and entries are updated and added to it by RE Loader every time events are processed for services belonging to the account that qualifies for out-of-order detection.

You can use the **purge_profile_event_ordering** script to clean up entries for closed billing cycles from the **/profile/event_ordering** object as follows:

Log on to the machine where **dm_oracle** is installed and run the Shell script:

```
BRM_Home/apps/pin_rel/purge_profile_event_ordering
```

The **purge_profile_event_ordering** script is located in *BRM_Home/apps/pin_rel*.

Note: Cleaning up data for closed billing cycles is not automatically synchronized with Pipeline Manager. You must restart Pipeline Manager for the in-memory entries to be cleaned up.

About Trigger-Dependent Rerating

Trigger-dependent rerating enables you to specify when events are automatically rerated. For example, if you rerate usage following a product cancellation, you can set up a product cancellation event to automatically trigger rerating.

To configure trigger-dependent rerating, you set up event notification to call a custom opcode that analyzes events to determine if rerating is required. For example, the criteria for rerating might be:

- If the event is a cancel event.
- If the cancel event for a product requires proration on cancellation.
- If rerating needs to occur to calculate proration for this product.

If the notification event triggers rerating, the custom opcode specifies which events for an account must be rerated and sends the rerating requirements to the PCM_OP_RERATE_INSERT_RERATE_REQUEST opcode to create a rerate job.

Trigger-dependent rerating works as follows:

1. An event that you have configured in event notification occurs to call the custom opcode. For example, this might be a purchase event, product cancellation, or change in account status. All the fields in the event are passed as input to the custom opcode.
2. The custom opcode analyzes the event using your custom selection criteria to determine if it should trigger rerating. For example, the opcode might be called when all purchase events occur, but not all purchase events will trigger rerating.
3. If the event triggers rerating because it matches your selection criteria, the custom opcode assigns an optional rerate reason code for rerating those events, specifies any price overrides, and calls PCM_OP_RERATE_INSERT_RERATE_REQUEST to create the rerate job.
4. PCM_OP_RERATE_INSERT_RERATE_REQUEST creates a rerate job that includes:
 - The account that needs to be rerated.
 - The events that must be rerated for that account.
 - The start time from which to rate the events.
 - The rerate reason.
 - Price overrides, if any.

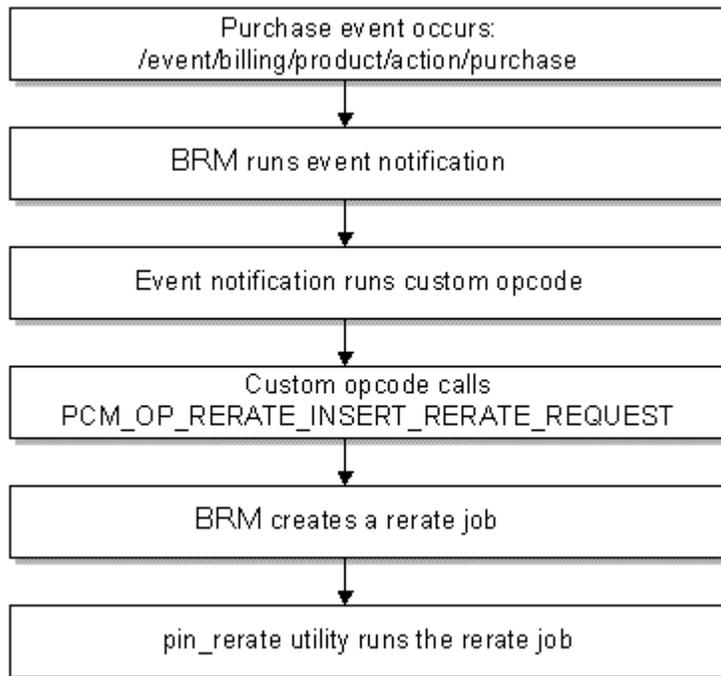
Note: If more than one account is included in the rerate job, the same selection criteria, price overrides, and start time apply to all the accounts.

For more information on how PCM_OP_RERATE_INSERT_RERATE_REQUEST creates rerate jobs, see "[How BRM Creates Rerate Jobs](#)".

5. You run the "[pin_rerate](#)" utility to execute the rerate job and rerate the events.

Figure 29–2 shows the trigger-dependent rerating process when a purchase event occurs:

Figure 29–2 Trigger-Dependent Rerating Process



To set up trigger-dependent rerating, see ["Setting Up Trigger-Dependent Rerating"](#).

Setting Up Trigger-Dependent Rerating

To set up trigger-dependent rerating, you do the following:

1. Create a custom opcode for trigger-dependent rerating. See ["Creating a Custom Opcode for Trigger-Dependent Rerating"](#).
2. Configure event notification for trigger-dependent rerating. See ["Configuring Event Notification for Trigger-Dependent Rerating"](#).

Creating a Custom Opcode for Trigger-Dependent Rerating

To use trigger-dependent rerating, you must write custom code to specify when to create rerate jobs automatically when certain events occur. You can write your code by doing one of the following:

- Create one or more custom opcodes.
You can create multiple custom opcodes to handle rerating events for different scenarios or create one opcode to handle all scenarios.
- Use the PCM_OP_SUBSCRIPTION_POL_GENERATE_RERATE_REQUEST policy opcode.
This policy opcode handles rerating events for BRM automatic rerating scenarios. You can modify and recompile this policy opcode to handle your own automatic rerating scenarios.

For more information about this opcode, see ["About Automatic Rerating"](#).

The opcode you use to define your custom code for trigger-dependent rerating must call the PCM_OP_RERATE_INSERT_RERATE_REQUEST opcode.

Your custom code is required to pass in the following to PCM_OP_RERATE_INSERT_RERATE_REQUEST:

- The POIDs of the accounts to be rerated.

You can also pass in an optional array of additional accounts that are related to these accounts (for example, an account that used the account's service before a line transfer) and the start time for each account.

Note: If an array of accounts is passed in, the same selection criteria and price overrides apply to all the accounts.

- The time from which events must be rerated for the accounts.

Your custom code can optionally pass in the following to PCM_OP_RERATE_INSERT_RERATE_REQUEST:

- A rerate reason code to take advantage of rerating according to type of rerate job. See "[Specifying a Rerate Reason Code](#)".
- Selection criteria so that only those events that are required for rerating are identified for an account. See "[Specifying Selection Criteria](#)".
- Price overrides to substitute an account's subscribed pricing for alternate pricing during rerating. See "[Specifying Price Overrides](#)".

Your custom selection criteria and price override information for creating rerate jobs can include anything that can be passed as input to the PCM_OP_RERATE_INSERT_RERATE_REQUEST opcode. For detailed information about the array and field names that can be passed, refer to the PCM_OP_RERATE_INSERT_RERATE_REQUEST input and output flist specifications and the class specifications of the `/job/rerate` and `/job_batch/rerate` storable classes.

When an event triggers your custom opcode, your opcode may need to obtain data from other events in the database to obtain all of the information required for rerating.

Specifying a Rerate Reason Code

PCM_OP_SUBSCRIPTION_POL_GENERATE_RERATE_REQUEST or your custom opcode can pass an optional rerate reason code to PCM_OP_RERATE_INSERT_RERATE_REQUEST. The rerate reason code is stored in the rerate job and can be used to select jobs from the rerate queue for rerating using `pin_rerate`.

Your rerate reason codes can be any integer, except 1 (1 is a reserved default value for pipeline-generated rerating requests).

Important: Because you can use multiple rerate reason codes to group rerate jobs for different rerating scenarios, ensure that your rerate reason codes are unique.

The default reason code is 0.

For information about assigning a rerate reason code by using PCM_OP_SUBSCRIPTION_POL_GENERATE_RERATE_REQUEST, see "[About Automatic Rerating](#)".

For information about processing rerate jobs according to the rerate reason code, see "[About Processing Rerate Jobs Created by Automatic Rerating](#)".

Specifying Selection Criteria

When an account needs to be rerated, not every event for that account may need to be rerated from the specified rerate start time. Your custom opcode can optionally send selection criteria to the PCM_OP_RERATE_INSERT_RERATE_REQUEST opcode so that all events for the account can be filtered to select only the subset of events that needs to be rerated.

For example, you may need to rerate events only when they are generated by a specific service or only when they are rated by a specific product or discount. You can specify selection criteria to rerate all accounts' events that have an event type of `/event/delayed/session/telco/gsm/telephony` but only when they are for the service type `/service/telco/gsm/telephony` and only when they are rated by the product GSM Voice Telephony.

Your selection criteria can be any of the following:

- Event types (array PIN_FLD_EVENTS)
- Service types (array PIN_FLD_SERVICES)
- Product POIDs (array PIN_FLD_PRODUCTS)
- Discount POIDs (array PIN_FLD_DISCOUNTS)
- Deal POIDs (array PIN_FLD_DEALS)

Note: Products and discounts are mutually exclusive with deals. Deals include products and discounts, so you can pass in a deal array instead of the product array and discount array.

Specifying Price Overrides

You can specify products, discounts, or deals to use for a specific rerate job to override an account's subscribed product, discount, or deals during rerating. The price override applies only to that rerate job; subsequent rerate jobs use the subscribed pricing or their own override pricing.

Your custom opcode can send the following price override data to the PCM_OP_RERATE_INSERT_RERATE_REQUEST opcode to set a price override for a rerate job:

- An optional array of override product POIDs along with the product it is overriding (PIN_FLD_OVERRIDE_PRODUCTS).
- An optional array of override discount POIDs along with the discount it is overriding (PIN_FLD_OVERRIDE_DISCOUNTS).
- An optional array of override deal POIDs along with the deal it is overriding (PIN_FLD_OVERRIDE_DEALS).

Note: Products and discounts are mutually exclusive with deals because deals include products and discounts. The deal is translated to a list of products and discounts when it is passed to the rerating opcodes.

The override products and discounts or deals must already be defined in your BRM database. They must also be functionally equivalent to the subscribed products and

discounts or deals; for example, the priority or event map would be the same. In addition, override products and discounts cannot be configured as first usage products. See "[About Effective Periods That Start on First Usage](#)" for information about first usage.

Important: You must rerate accounts when an override pricing product is canceled in a given billing cycle so that refunds can be applied correctly. See "[Configuring Event Notification for Override Pricing](#)".

BRM creates a record when rerating is run with price overrides by using the `/rerate_session/override_info` object. When override pricing is passed to the `PCM_OP_SUBSCRIPTION_RERATE_REBILL` opcode, this opcode creates a `/rerate_session/override_info` object to capture the override pricing information used during rerating.

Price overrides are automatic when you use the Best Pricing feature. For that feature, BRM calculates the best price from alternate products and discounts and rerates events to use the best pricing. When rerating events, the product or discount that provides the best pricing is the override product or discount.

Configuring Event Notification for Trigger-Dependent Rerating

When you configure event notification for trigger-dependent rerating, you specify the following in the `pin_notify` file in `BRM_Home/sys/data/config` (or your own merged event notification configuration file):

- The events that trigger rerating.
- The custom opcodes you want BRM to use to analyze the events, to identify whether rerating is required.

To configure event notification for trigger-dependent rerating, do the following:

1. If your system has multiple configuration files for event notification, merge them.
2. Ensure that the merged file includes the events you want to trigger rerating and the opcode number of the custom opcode you want to analyze those events.

For example, these entries call the custom opcode with opcode number 10000:

```
# Event notification for trigger-dependent rerating
10000 0 /event/customer/status
10000 0 /event/billing/product/action/purchase
```

3. (Optional) Add, modify, or delete entries in your final event notification list.
4. (Optional) Create custom code for event notification to trigger.
5. Load your final event notification list into the BRM database.

For more information, see "Using Event Notification" in *BRM Developer's Guide*.

Configuring Event Notification for Override Pricing

If you use override pricing, you must set up trigger-dependent rerating to rerate accounts for cases where a refund could not be applied because an override pricing product was canceled in a given billing cycle.

For a given billing cycle, if you rerate an account with an override product and the product is canceled, BRM cannot calculate a refund for that account because the

product is not available to apply the necessary cycle fees. When the refund cannot be applied, BRM creates the notification event **/event/notification/product/cancel/no_refund**.

To calculate the correct refund amount, you must set up trigger-dependent rerating as follows:

1. Write custom code to specify:
 - The account to rerate from the **/event/notification/product/cancel/no_refund** event.
 - The product to use to apply the cycle fees for the refund to use the override product that was canceled.

BRM uses the base pricing product associated with the account if you do not specify the override product that was canceled. To specify the override product that was canceled, obtain its information from the latest **/rerate_session/override_info** object created for the given account.

2. Include your code in a custom opcode or in the **PCM_OP_SUBSCRIPTION_POL_GENERATE_RERATE_REQUEST** policy opcode that handles automatic rerating (opcode number 3787).

If you create a custom opcode, it must call the **PCM_OP_RERATE_INSERT_RERATE_REQUEST** opcode to create the rerate job.

3. Set up event notification for the event **/event/notification/product/cancel/no_refund** to call your custom opcode or the policy opcode. For example, to call the **PCM_OP_SUBSCRIPTION_POL_GENERATE_RERATE_REQUEST** policy opcode, you would enter:

```
# Event notification for trigger-dependent rerating
3787 0 /event/notification/product/cancel/no_refund
```

For detailed instructions on setting up event notification, see "[Configuring Event Notification for Trigger-Dependent Rerating](#)".

Using the pin_rerate Utility

This chapter provides an overview of how you use the Oracle Communications Billing and Revenue Management (BRM) **pin_rerate** utility and the functions the utility performs.

For general information about rerating, see ["About Rerating Events"](#).

For information about what you can do with **pin_rerate**, see ["About Comprehensive Rerating Using pin_rerate"](#).

About Using the pin_rerate Utility

You use the **pin_rerate** command-line utility to perform the following tasks:

- Select accounts and events for rerating from the BRM database. When accounts are selected, **pin_rerate** creates rerate jobs for the selected accounts. See ["Selecting Accounts and Events for Rerating"](#).

You can assign a rerate reason code to the jobs created for the selected accounts and events. This enables you to rerate the accounts separately based on the reason for rerating. See ["Assigning Rerate Reasons to Rerate Jobs"](#).

You can also define custom **pin_rerate** parameters based on various event criteria. This enables you to further customize which event attributes are used to select accounts and events for rerating. See ["Defining Custom pin_rerate Parameters for Rerating"](#).

- Rerate the accounts. To rerate accounts, you process rerate jobs. The process for rerating depends on whether you rerate real-time-rated and pipeline-batch-rated events concurrently or, if you do not use Pipeline Manager for batch rating, rerate only real-time-rated events. See the following sections:
 - [Rerating Real-Time-Rated and Pipeline-Rated Events Concurrently](#)
 - [Processing Rerate Jobs When You Do Not Use Pipeline Batch Rating](#)

For information about processing jobs that are automatically created by BRM automatic rerating features, see ["About Processing Rerate Jobs Created by Automatic Rerating"](#).

Important: Do not move accounts to another database schema while rerating events for those accounts.

- Reprocess any rerate jobs that failed. See ["Processing Failed Rerate Jobs"](#).

- Back out the balance impacts of rating without rerating the events. See "[Using pin_rerate for Back-Out-Only Rerating](#)".
- Generate reports about the results of rerating. See "[Reports Generated by the pin_rerate Utility](#)".

If rerating fails, **pin_rerate** creates a report that includes the account numbers and start times for failed rerate jobs. The report file name is **pin_rerate.status_report** and is in the directory from which you ran the utility.

When rerate jobs have been processed, you can run **pin_rerate** to purge them from the database. See "[Parameter for Purging Rerate Jobs](#)".

Selecting Accounts and Events for Rerating

The first step in the rerating process is running **pin_rerate** to select the accounts and associated events to rerate from the BRM database. The **pin_rerate** utility creates rerate jobs (**/job/rerate** and **/job_batch/rerate** objects) to store the information about the selected accounts. See "[About Rerate Jobs](#)".

Note: If you use automatic rerating, accounts are selected and rerate jobs are created automatically for certain scenarios. For more information, see "[About Automatic Rerating](#)".

Caution: Do not move accounts to another database schema while rerating events for those accounts.

Specifying Accounts for Rerating

By default, the **pin_rerate** utility selects for rerating all the accounts and then all the events associated with those accounts that occurred from the start time that you specify.

For example, the following command selects all the accounts from the BRM database and, for those accounts, selects all the events that occurred after 07/23/2007.

```
pin_rerate -t 07/23/2007
```

pin_rerate provides a set of parameters that you can optionally use to select only specific accounts that meet one of the following requirements:

- One or a set of accounts identified by the account POIDs
- Accounts with events rated by a particular product
- Accounts with events rated by a particular discount
- Accounts with events rated by products and discounts associated with a particular deal
- Accounts with events generated for a particular service type or subscription service
- Accounts with events associated with an account's bill unit or bill unit and balance group
- Accounts that have particular event types

For example, when you use the **-p** parameter:

```
pin_rerate -p products.txt -t 07/23/2007
```

pin_rerate does the following:

- Selects only the accounts that have events related to the products in the **products.txt** file.
- Selects all the events for the selected accounts that occurred after 07/23/2007.

When you use the **-line** parameter:

```
pin_rerate -line 6495832245 -t 09/21/04
```

pin_rerate does the following:

- Selects only the accounts that have the subscription service with the phone number 6495832245.
- Selects all the events for the selected accounts that occurred after 09/21/04.

For a complete list of the **pin_rerate** parameters, see "[pin_rerate](#)".

Specifying Events for Rerating

By default, **pin_rerate** rerates all the events for the selected accounts from the start time that you specify. You can specify to rerate only specific events for the selected accounts by using the **-r** parameter. This is called *selective rerating*.

Important: Do not use selective rerating if:

- Your rating scheme includes credit limits or resource-based tiers. These schemes require that all events related to an account are rerated to assure accurate rerating.
 - Deferred taxation is used for taxing events during rerating.
-

If you use selective rerating, be sure to consider how it might affect rating overall because account balances can be impacted by different types of events: a cycle event can grant free minutes and a usage event consumes free minutes from the same balance. If, for example, you change the pricing for a product that grants free minutes, you must rerate all events for the accounts that own the product. It would be incorrect to use selective rerating in this case.

The **-r** parameter must be used with parameters that specify the accounts to select for rerating. When the **-r** parameter is used, rerating applies the account selection criteria to the account's events as well and selects only those events that meet the selection criteria.

The **-r** parameter can be used with any of the account selection parameters.

For example, when you use **-r** with the **-s** parameter:

```
pin_rerate -r -s service.txt -t 07/23/2007
```

pin_rerate does the following:

- Selects only the accounts that have events related to the services in the **service.txt** file.
- Selects only the events related to the services in **service.txt** that occurred after 07/23/2007 for the selected accounts.

When you have a high volume of events to rerate, you can rerate events that are rated only in real time, such a cycle and purchase events. To do this, you use the **-r** parameter in combination with the **-n** parameter for specifying event types. You define all the event types rated by real-time rating in an input file.

For example, you could specify the following event types in a file named **event.txt**:

- **/event/billing/product/fee/cycle/cycle_forward_monthly**
- **/event/billing/product/fee/purchase**
- **/event/billing/product/fee/cycle/cycle_forward_arrear**

When you run the following command:

```
pin_rerate -t 01/01/2007 -n event.txt -r
```

pin_rerate does the following:

- Selects all accounts that have events with event types in the **event.txt** file.
- Selects only the events with event types in the **event.txt** file and which occurred after 01/01/2007 for the selected accounts.

When rerating cycle fee events, to get the correct rerating results, include the cycle events that occur during billing that are configured in the product, such as cycle discount, rollover, and fold events in the event file.

For example, if a cycle discount is configured to be some percentage of the charge during billing and if the cycle forward arrear fee is modified during the billing cycle, then to rerate the cycle forward arrear event, you need to include both events in the event file:

- **/event/billing/product/fee/cycle/cycle_forward_arrear**
- **/event/billing/cycle/discount**

If the event type specified in the **-n** parameter input file has subclasses, all subclass events are also rerated, providing they meet the selection criteria. For example, if you specify **/event/delayed/session/telco** in the **-n** parameter input file, events of type **/event/delayed/session/telco/gsm** that meet the selection criteria are also rerated.

For a complete list of the **pin_rerate** parameters, see "[pin_rerate](#)".

Customizing Event Searches for Selective Rerating

You can further refine the event selection criteria used for selective rerating by customizing the **PCM_OP_SUBSCRIPTION_POL_SPEC_RERATE** policy opcode. This opcode is called when the **pin_rerate** utility is run with **-r** parameter to indicate selective rerating.

Note: An alternative to customizing this policy opcode to filter the events selected for rerating is to create custom **pin_rerate** parameters instead. See "[Defining Custom pin_rerate Parameters for Rerating](#)".

PCM_OP_SUBSCRIPTION_POL_SPEC_RERATE receives an event search template that is based on the account and event selection criteria specified in the **pin_rerate** command line: for example, to select events related to services specified by the **-s** parameter.

By default, **PCM_OP_SUBSCRIPTION_POL_SPEC_RERATE** does not change the search template and returns a copy of the input flist in the output flist.

You can customize the search template in the input flist to rerate specific types of events. Most customizations include changes only to the fields listed in [Table 30–1](#):

Table 30–1 Fields to Customize

Field	Description
PIN_FLD_TEMPLATE	The modified search template.
PIN_FLD_ARGS	The list of search arguments. Note: <ul style="list-style-type: none"> ▪ This list must match the list of arguments in PIN_FLD_TEMPLATE. ▪ It is preferable to have arguments of one type only. For example, an event search based on product objects.

The PCM_OP_SUBSCRIPTION_POL_SPEC_RERATE policy opcode receives the following fields in the RESULTS array from the PCM_OP_SUBSCRIPTION_RERATE_REBILL standard opcode:

- PIN_FLD_POID
- PIN_FLD_CREATED_T
- PIN_FLD_EFFECTIVE_T
- PIN_FLD_END_T
- PIN_FLD_SERVICE_OBJ
- PIN_FLD_ACCOUNT_OBJ
- PIN_FLD_UNRATED_QUANTITY
- PIN_FLD_RERATE_OBJ
- PIN_FLD_BAL_IMPACTS [*]
- PIN_FLD_SUB_BAL_IMPACTS [*]

Important:

- To assure that the existing mandatory fields in the array are passed back, avoid customizing the RESULTS array. Extra fields added to the array are retrieved but ignored by the standard opcode.
 - Test your customized template to ensure that it works properly before using it with a working database. Use the **-e** and **-c** parameters with the **pin_rerate** utility to test your modifications. See "[pin_rerate](#)".
-
-

The PCM_OP_SUBSCRIPTION_POL_SPEC_RERATE policy opcode returns the search template that the PCM_OP_SUBSCRIPTION_RERATE_REBILL opcode uses to find events in selected accounts that need rerating.

Specifying the Event Sequence for Rerating

Events are rerated in sequence based on the event time. You can specify one of two times:

- *Event end time* defines the time that the usage actually occurred. This is the default.
- *Event creation time* is the time that the event was loaded into the BRM database.

The event time you specify might depend on how the original events were rated:

- **Events rated or loaded in batches**

Events that are rated or loaded into the BRM database in batches have a significant delay between the event end time and creation time. Using the event end time reflects the actual real-time sequence of the original events. However, because batch events are recorded in order of creation time, this makes predicting the actual impact of a price configuration change harder. To compare the original and rerated balance impacts of batch events, use the event creation time.

- **Events rated and loaded in real time**

Events that are rated and loaded in real-time have very little delay between the event end time and creation time. Both the event end time and creation time reflect the real-time sequence in which the original events occurred and were recorded.

By default, the **pin_rerate** utility rerates batch events in sequence based on event end time for both real-time and batch events.

To specify the event time for rerating, use the **-b** parameter when selecting accounts for rerating. The **-b** parameter takes either the **c** option (for event creation time) or the **e** option (for event end time).

For example:

```
pin_rerate -b c -t 07/23/2007
```

The preceding example selects all accounts, then selects both real-time and batch events that occurred after 07/23/2007 for the selected accounts, and finally rerates the events in order of creation time.

For a list of the **pin_rerate** parameters, see "[pin_rerate](#)".

Assigning Rerate Reasons to Rerate Jobs

Some rerating jobs must be processed before others. For example, if a rerate job includes events that have an impact on future rating, such as volume-based discounting, those events must be rerated first. You can achieve this by assigning rerate reason codes to rerate jobs when you create those jobs. You can then select only those jobs associated with the specified rerate reason code during scheduled rerating executions.

Rerate reason codes can be any integer except **1** (**1** is reserved for pipeline-generated events that were rated out of order).

Important: Ensure that your reason codes are unique to process rerate jobs associated with them during separate rerating executions.

You assign rerate reason codes when you manually create rerate jobs by using the **pin_rerate** utility.

Note: Rerate jobs created automatically by BRM automatic rerating features are not assigned a reason code by default. You can assign rerate reason codes for trigger-dependent and automatic rerating by customizing the PCM_OP_SUBSCRIPTION_POL_GENERATE_RERATE_REQUEST policy opcode. For more information, see the following sections:

- [Setting Up Trigger-Dependent Rerating](#)
- [About Automatic Rerating](#)

You can assign a unique reason code for out-of-order rerating by using a configuration file. See "[Specifying a Reason Code for Rerating Out-of-Order Events](#)".

To assign a rerate reason code by using **pin_relate**, you specify the rerate reason code on the command line along with other account selection parameters, such as the start time and other parameters:

```
pin_relate -reason reason_code -t start_time other_parameters
```

You can specify only one reason code when creating rerate jobs. Rerate jobs are created for the selected accounts and all rerate jobs are assigned the specified rerate reason code.

For example, to assign a rerate reason code of **99** to rerate jobs that rerate all account's events that occurred after January 1, 2007 and that are associated with the products specified in the **product.txt** file, you enter the following command:

```
pin_relate -reason 99 -t 01/01/2007 -p product.txt
```

For more information, see "[pin_relate](#)".

To process rerate jobs according to the rerate reason code, see the following sections:

- [Rerating Real-Time-Rated and Pipeline-Rated Events Concurrently](#)
- [Processing Rerate Jobs When You Do Not Use Pipeline Batch Rating](#)

Defining Custom pin_relate Parameters for Rerating

You can define custom **pin_relate** parameters based on any event criteria. This enables you to customize which event attributes are used to select accounts and events for rerating.

To define custom parameters, you map extraction keys to fields in the event object. You then run the **load_pin_relate_flds** utility to load the extraction-key-to-event-field mappings into the **/config/rerate_flds** object in the BRM database. See "[Configuring Custom pin_relate Parameters](#)".

When you run **pin_relate**, it uses the extraction key to find the corresponding event field name in the **/config/rerate_flds** object. It uses the event field name to find and retrieve accounts and events for rerating.

For example, if you map the extraction key **resource_id** to the PIN_FLD_RESOURCE_ID field in the **/event** object, you can run **pin_relate** with the following command, specifying the input file containing the resource IDs:

```
pin_relate -resource_id input_file -t start_time
```

In this example, **pin_rerate** selects accounts to be rerated that have events associated with the resources specified in the input file.

By default, BRM searches only the base **/event** class for fields associated with custom **pin_rerate** parameters. If the event field associated with a custom parameter is present only in a subclass, you must specify the subclass event type in the command line by including the **-n** parameter.

Important: When used with custom **pin_rerate** parameters, the input file for the **-n** parameter can contain only one event type. If you specify more than one event type, an error is returned.

Configuring Custom pin_rerate Parameters

To configure custom **pin_rerate** parameters, you perform the following tasks:

- [Defining Custom Parameters](#)
- [Loading Custom Parameters](#)

Defining Custom Parameters

To define custom **pin_rerate** parameters, create an XML file that maps the parameters to event fields. You can map to fields specified in the **EVENT_T** and **EVENT_BAL_IMPACTS_T** tables in the base **/event** class or to fields in an event subclass.

Note:

- If you map parameters to fields in an **/event** subclass, you must specify the subclass by using the **-n** parameter when you run **pin_rerate** with the custom parameter.
 - To create the XML file, you must be familiar with XML and the XML schema.
-
-

Create an XML file containing the parameter-to-event field mappings according to the BRM rerating XML schema file (*BRM_home/xsd/pin_rerate_flds.xsd*).

Important: Validate your XML file against the XML schema. The **load_pin_rerate_flds** utility cannot successfully load an invalid XML file.

Sample XML parameter file

The following example shows how you specify event fields in **<Name>** tags and the parameters they map to in **<value>** tags:

```
<PinRerateList xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.portal.com/InfranetXMLSchema pin_rerate_flds.xsd">
  <PinRerate>
    <Name>EVENT.PIN_FLD_START_T </Name>
    <value>startTime</value>
  </PinRerate>
  <PinRerate>
    <Name>EVENT.PIN_FLD_END_T</Name>
    <value>endTime</value>
  </PinRerate>
```

```

<PinRerate>
  <Name>EVENT.PIN_FLD_ACCOUNT_OBJ</Name>
  <value>account</value>
</PinRerate>
<PinRerate>
  <Name>EVENT.PIN_FLD_RATE_PLAN </Name>
  <value>ratePlan</value>
</PinRerate>
<PinRerate>
  <Name>EVENT.PIN_FLD_DEAL </Name>
  <value>deal</value>
</PinRerate>
<PinRerate>
  <Name>EVENT.PIN_FLD_SERVICE_OBJ </Name>
  <value>service</value>
</PinRerate>
<PinRerate>
  <Name>EVENT.PIN_FLD_BAL_IMPACTS.PIN_FLD_PRODUCT_OBJ</Name>
  <value>product</value>
</PinRerate>
</PinRerateList>

```

If you map a custom parameter to a field in an `/event` subclass, specify the array or substruct (if any) that contains the parameter field.

For example, to specify `PIN_FLD_ORIGIN_NETWORK`, which is located in the `PIN_FLD_TELCO_INFO` substruct in the `/event/delayed/session/telco` subclass:

```

<PinRerate>
<Name>EVENT.PIN_FLD_TELCO_INFO.PIN_FLD_ORIGIN_NETWORK</Name>
<value>origin_network</value>.
</PinRerate>

```

Loading Custom Parameters

Use the following command to load the custom `pin_rerate` parameters defined in the XML file into the `/config/rerate_fld` object:

```
BRM_home/sys/data/config/load_pin_rerate_flds xml_file_name
```

Important: The `load_pin_rerate_flds` utility uses a configuration file (`pin.conf`) located in the same directory to connect to the BRM database. Edit the configuration file to connect to your BRM database.

For more information, see "[load_pin_rerate_flds](#)".

About Processing Rerate Jobs Created by Automatic Rerating

Rerate jobs are created automatically by the following features:

- Automatic rerating. See "[About Automatic Rerating](#)".
- Trigger-dependent rerating. See "[About Trigger-Dependent Rerating](#)".
- Out-of-order rerating. See "[About Automatic Rerating of Out-of-Order Events](#)".

These rerate jobs are automatically processed when you run the `pin_rerate` with one of the following parameters:

- The **-process jobs** parameter when rerating real-time-rated and pipeline-rated events concurrently.
- The **-rerate** parameter when rerating only real-time-rated if you do not use Pipeline Manager for batch rating.

By default, rerate jobs that are automatically created are not assigned a rerate reason code. If you customize automatic rerating to assign a specific rerate reason code to rerate jobs, you process those jobs by using the **-reason** parameter with the **-process jobs** or **-rerate** parameter.

For more information, see the following sections:

- [Rerating Real-Time-Rated and Pipeline-Rated Events Concurrently](#)
- [Processing Rerate Jobs When You Do Not Use Pipeline Batch Rating](#)

Rerating Real-Time-Rated and Pipeline-Rated Events Concurrently

When rerating real-time-rated and pipeline-rated events concurrently, you process rerate jobs in the following ways:

- By processing all existing rerate jobs that were previously created. See "[Processing Rerate Jobs for Concurrent Rerating](#)".
- By processing only rerate jobs associated with a rerate reason code. See "[Processing Rerate Jobs According to a Rerate Reason Code for Concurrent Rerating](#)".

Important: Ensure that no event data records (EDRs) are in the pipeline before you run **pin_rerate**. Otherwise, EDRs might be rated using old account data.

Processing Rerate Jobs for Concurrent Rerating

When rerating real-time-rated and pipeline-batch-rated events concurrently, rerate jobs are created and processed in separate commands. Rerate jobs can be created in the following ways:

- Automatically by BRM automatic rerating features.
- When you run **pin_rerate** to select accounts for rerating, as described in "[Selecting Accounts and Events for Rerating](#)". For example:

```
pin_rerate -t start_time -s input_file
```

To process all existing rerate jobs in the rerate queue, run the following commands:

1. **pin_rerate -process jobs**. See "[Notifying the Batch Pipeline That Rerating Is Starting](#)".
2. **pin_rerate -process queue**. See "[Processing Acknowledgment Events from the Batch Pipeline](#)".
3. **pin_rerate -rerate**. See "[Rerating the Events Associated with the Accounts in Rerate Jobs](#)".
4. **pin_rerate -process queue**. See "[Processing Acknowledgment Events from the Batch Pipeline](#)".
5. **pin_rerate -process recycle**. See "[Recycling EDRs Suspended during Rerating](#)".

For example:

```
pin_rerate -process jobs
pin_rerate -process queue
pin_rerate -rerate
pin_rerate -process queue
pin_rerate -process recycle
```

Note: You run **pin_rerate** during non-peak hours with these commands as part of a cron job. See your operating system documentation for details on creating a cron job.

Notifying the Batch Pipeline That Rerating Is Starting

After rerate jobs are created, you run **pin_rerate** with the **-process jobs** parameter to notify the batch pipeline that rerating is about to begin.

When you specify the **-process jobs** parameter, the following actions are performed:

1. The **pin_rerate** utility finds all the rerate jobs with a status of NEW and sends business events containing the rerate job ID and the list of accounts associated with the rerate job to the batch pipeline.
2. The **pin_rerate** utility then changes the status of the jobs from NEW to WAITING_ACCOUNT_LOCKED to indicate that it is waiting for an acknowledgment from the pipeline.
3. The batch pipeline dequeues the business events and suspends rating EDRs for those accounts. See "[About Suspending and Recycling EDRs during the Rerating Process](#)".
4. The batch pipeline sends an acknowledgment event back to the Account Synchronization database queue.

Note:

- If you process rerate jobs according to a rerate reason, the same actions are performed but only for the rerate jobs associated with the rerate reasons you specify.
 - To process rerate jobs according to a rerate reason by using the **pin_rerate** utility, see "[Processing Rerate Jobs According to a Rerate Reason Code for Concurrent Rerating](#)".
-

Processing Acknowledgment Events from the Batch Pipeline

You run **pin_rerate** with the **-process queue** parameter to process acknowledgment events sent by the batch pipeline.

You run **pin_rerate** with the **-process queue** parameter at two times:

- After you run **pin_rerate** with the **-process jobs** parameter. The acknowledgment event at this time indicates that the batch pipeline has suspended rating EDRs for the accounts in the rerate jobs. The **pin_rerate** utility dequeues the event and changes the status of the rerate jobs from WAITING_ACCOUNT_LOCKED to ACCOUNT_LOCKED.
- After you run **pin_rerate** with the **-rerate** parameter. The acknowledgment event at this time indicates that the batch pipeline has resumed rating EDRs for the

accounts in the rerate job. **pin_rerate** dequeues the event and changes the status of the rerate jobs from RERATED to READY_FOR_RECYCLE.

Rerating the Events Associated with the Accounts in Rerate Jobs

When the rerate jobs has the ACCOUNT_LOCKED status, you run **pin_rerate** with the **-rerate** parameter to rerate the accounts.

pin_rerate finds all the rerate jobs with a status of ACCOUNT_LOCKED and calls rerating opcodes to rerate the accounts.

After an account is successfully rerated, **pin_rerate** sends a business event through the Account Synchronization Data Manager (DM) to the batch pipeline to update the account data. The DAT_BalanceBatch module processes the event and reloads the updated account balances from the BRM database.

When *all* the accounts in a rerate job have been successfully rerated and updated, **pin_rerate** updates the rerate job status from ACCOUNT_LOCKED to RERATED and then notifies the batch pipeline that rerating is complete. The batch pipeline resets the rerate flags for all the accounts in the rerate job and returns an acknowledgment event to inform **pin_rerate** that it has resumed rating EDRs for those accounts.

Recycling EDRs Suspended during Rerating

EDRs that are temporarily suspended by the batch pipeline are loaded into the BRM database by Suspended Event (SE) Loader. The suspended EDRs are stored in the database until they are recycled.

Important: Before you can recycle suspended EDRs, they must be loaded into the BRM database by SE Loader. You typically schedule SE Loader to run automatically when you set up standard recycling.

To recycle the EDRs that were suspended during the rerating process, you run **pin_rerate** with the **-process recycle** parameter.

Note: Suspended EDRs are typically recycled by running the **pin_recycle** utility. However, **pin_rerate** calls the standard recycling opcode directly so you do not use **pin_recycle** when using **pin_rerate**.

pin_rerate finds all the rerate jobs with a status of READY_FOR_RECYCLE and calls the standard recycling opcodes to recycle the associated EDRs. Standard recycling uses the recycle key value in the EDR to identify and retrieve the EDRs that were suspended during the rerating process.

After the EDRs are recycled, **pin_rerate** changes the status of the jobs from READY_FOR_RECYCLE to COMPLETE.

Note:

- In a multischema environment, you must run `pin_rerate` separately on each database schema.
- If no EDRs were suspended for the accounts being rerated, the job status is still changed to COMPLETE.
- If an error occurs while recycling EDRs, the job status is not changed; it retains the status of READY_FOR_RECYCLE.

Processing Rerate Jobs According to a Rerate Reason Code for Concurrent Rerating

To process rerate jobs according to a rerate reason code when rerating real-time-rated and pipeline-batch-rated events concurrently, you use the `-process jobs` parameter with the `-reason` parameter:

```
pin_rerate -process jobs -reason [reason_code_1,reason_code_2,reason_code_3,...]
```

where `reason_code_x` is the rerate reason code assigned to existing rerate jobs in one of the following ways:

- When you run `pin_rerate` with the `-reason` parameter to create rerate jobs and with an assigned rerate reason code. See ["Assigning Rerate Reasons to Rerate Jobs"](#).
- When you customized one of the following features to assign a specific rerate reason code to rerate jobs that are automatically created:
 - Automatic rerating. See ["About Automatic Rerating"](#).
 - Trigger-dependent rerating. See ["About Trigger-Dependent Rerating"](#).
 - Out-of-order rerating.

You can list multiple reason codes separated by commas (do not use spaces between reason codes). For example, to process jobs with reason codes 100, 200, and 300, enter the following commands:

```
pin_rerate -process jobs -reason 100,200,300
pin_rerate -process queue
pin_rerate -rerate
pin_rerate -process queue
pin_rerate -process recycle
```

Important: When you list multiple rerate reasons, the rerate jobs associated with them are processed randomly. They are not processed in the order you list them in the command line, nor are they processed in increasing or decreasing numeric order. To process rerate jobs according to a rerate reason in a particular order, you must schedule separate rerating executions for them.

Processing Rerate Jobs When You Do Not Use Pipeline Batch Rating

If you do not use Pipeline Manager for batch rating, you process rerate jobs in the following ways:

- By creating rerate jobs and processing those jobs at the same time. See ["Processing Rerate Jobs When Selecting Accounts for Rerating"](#).

- By processing all existing rerate jobs that were previously created. See "[Processing Existing Rerate Jobs](#)".
- By processing only rerate jobs associated with a rerate reason code. See "[Processing Rerate Jobs per a Rerate Reason When Rerating Only Real-Time-Rated Events](#)".

Processing Rerate Jobs When Selecting Accounts for Rerating

When rerating real-time events only, rerate jobs are processed when you run **pin_rerate** and specify the start time and any additional search criteria as described in "[Selecting Accounts and Events for Rerating](#)".

For example:

```
pin_rerate -t start_time -p input_file
```

The **pin_rerate** utility selects the accounts for rerating by using the search criteria, creates rerate jobs, and then immediately rerates the selected accounts.

Processing Existing Rerate Jobs

Some rerate jobs are automatically created by BRM automatic rerating features. Other rerate jobs can be precreated by assigning a rerate reason code when selecting the accounts for rerating.

To process all existing rerate jobs in the rerate queue, you use only the **-rerate** parameter:

```
pin_rerate -rerate
```

Processing Rerate Jobs per a Rerate Reason When Rerating Only Real-Time-Rated Events

To process rerate jobs according to a rerate reason when rerating only real-time-rated events, you use the **-rerate** parameter with the **-reason** parameter:

```
pin_rerate -rerate -reason reason_code_1,reason_code_2,reason_code_3,...
```

where *reason_code_x* is the rerate reason code assigned to existing rerate jobs in one of the following ways:

- When you run **pin_rerate** with the **-reason** parameter to create rerate jobs with an assigned rerate reason code. See "[Assigning Rerate Reasons to Rerate Jobs](#)".
- When you customized one of the following features to assign a specific rerate reason code to rerate jobs that are automatically created:
 - Automatic rerating. See "[About Automatic Rerating](#)".
 - Trigger-dependent rerating. See "[About Trigger-Dependent Rerating](#)".
 - Out-of-order rerating. See "[About Automatic Rerating of Out-of-Order Events](#)".

You can list multiple reason codes separated by commas (do not use spaces between reason codes). For example, to process jobs with reason codes 100, 200, and 300, you enter the following commands:

```
pin_rerate -rerate -reason 100,200,300
```

Important: When you list multiple rerate reasons, the rerate jobs associated with them are processed randomly. They are not processed in the order you list them in the command line, nor are they processed in increasing or decreasing numeric order. To process rerate jobs according to a rerate reason in a particular order, you must schedule separate rerating executions for them.

Processing Failed Rerate Jobs

To process failed rerate jobs, you run **pin_rerate** only with the commands for processing rerate jobs, without specifying selection criteria or rerate reason codes.

Note: This also processes all rerate jobs that are in the rerate queue: not only failed rerate jobs.

For example:

- When rerating real-time-rated and pipeline-rated events concurrently, use the following commands:

```
pin_rerate -process jobs
pin_rerate -process queue
pin_rerate -rerate
pin_rerate -process queue
pin_rerate -process recycle
```

See "[Rerating Real-Time-Rated and Pipeline-Rated Events Concurrently](#)".

- If you do not use Pipeline Manager for batch rating, use the following command:

```
pin_rerate -rerate
```

See "[Processing Rerate Jobs When You Do Not Use Pipeline Batch Rating](#)".

For more information, see "[How Failed Rerate Jobs Are Processed](#)".

Using pin_rerate for Back-Out-Only Rerating

Back-out-only rerating backs out the balance impacts of rating without rerating events.

BRM backs out balance impacts for back-out-only rerating by creating an adjustment event that fully negates the original balance impacts. See "[How BRM Applies the Balance Impacts of Rerating](#)".

To back out the balance impacts of rating, run **pin_rerate** with the **-backout** parameter. Use the **-backout** parameter with other parameters that select the accounts and their events for rerating. This creates rerate jobs that are set for back-out-only rerating for the selected accounts.

For example, the following command selects accounts whose events were rated by the products specified in the **products.txt** file and backs out those events rated by the products that occurred after 06/01/2007:

```
pin_rerate -backout -r -p products.txt -t 06/01/2007
```

The **-r** parameter is used to select only the events that used the specified products. Without the **-r** parameter, **pin_rerate** would select accounts whose events used the specified products and then back out all events for those accounts.

Note: Use caution when choosing the events to back out because it can impact your general ledger. For example, it is incorrect to use back-out-only rerating for a cycle event when the customer has already paid the cycle fee or to use back-out-only rerating when product pricing is changed. Typically, back-out-only rerating is performed only on usage events where rating should not have occurred.

Using Custom pin_rerate Parameters with Back-Out-Only Rerating

You can perform back-out-only rerating using custom **pin_rerate** parameters. This enables you to select events for back-out-only rerating based on any event criteria.

For information about defining custom **pin_rerate** parameters, see ["Defining Custom pin_rerate Parameters for Rerating"](#).

By default, rerating searches only the base **/event** storable class for fields associated with custom **pin_rerate** parameters. If the event field associated with a custom parameter is present only in a subclass, you must also specify the subclass event type in the command line by using the **-n input_file** parameter.

If the event type specified in the **-n** parameter input file has subclasses, all subclass events are also backed out, providing they meet the selection criteria. For example, if you specify **/event/delayed/session/telco** in the **-n** parameter input file, events of type **/event/delayed/session/telco/gsm** that meet the selection criteria are also backed out.

Example of Using a Custom Parameter with Back-Out-Only Rerating

Suppose you define the custom parameter **origin_network** to select events based on the **PIN_FLD_ORIGIN_NETWORK** field. You specify an origin network ID in a file named **origin_network.txt**.

The **PIN_FLD_ORIGIN_NETWORK** field is located in the **/event/delayed/session/telco** subclass so you specify this subclass, in a file named **event_subclass.txt**.

To select accounts whose events were generated by the specified origin network and back out the balance impacts of only those events when they occurred after 06/01/2007, run the following command:

```
pin_rerate -backout -r -n event_subclass.txt -origin_network origin_network.txt -t 06/01/2007
```

Reports Generated by the pin_rerate Utility

By default, the rerating process generates summary and detailed reports. To print a report, use the **pin_rerate -I** parameter. You have the option of printing either a summary of the report, a detailed report, or both summary and detailed reports. If you do not specify which report to print, both summary and detailed reports are printed by default.

You set the reporting parameter as follows:

- When rerating both real-time-rated and pipeline-batch-rated events concurrently, you specify the **-I** parameter with the **-rerate** parameter. In the following example, the report option is set to print a summary report:

```
pin_rerate -t 01/01/2007 -s service.txt
pin_rerate -process jobs
```

```
pin_rerate -process queue
pin_rerate -rerate -l s
pin_rerate -process queue
pin_rerate -process recycle
```

Important: When the batch pipeline is enabled, you cannot specify the report option at rerate job creation time.

- If you do not use Pipeline Manager for batch rating, you specify the **-l** parameter at one of the following times:

- When specifying the accounts to rerate. Enter the **-l** parameter after the parameters used to select the accounts. In the following example, the report option is set to print a detailed report:

```
pin_rerate -t 01/01/2007 -p product.txt -l d
```

- When processing rerate jobs that already exist, such as those created with a rerate reason code and those created by automatic rerating. Specify the **-l** parameter with the **-rerate** parameter. In the following example, the report option is set to print a summary report:

```
pin_rerate -rerate -l s
```

Important: Report generation is resource intensive and can degrade system performance, depending on system load and the number of events rerated. You can override report generation by using the **-l n** option with **-rerate** to skip printing reports. For example: **pin_rerate -rerate -l n**

Report Generated When Rerating Is Performed before Billing

Summary reports are created for each account. The following example of a summary report shows that rerating occurred as a result of changing the monthly cycle forward fee from \$200 to \$20. Rerating took place *before* billing was run, so the difference is shown in the current bill.

```

PIN_RERATE SUMMARY REPORT
=====

Date Range: 3/1/2007 10:0:0 to 3/2/2007 20:4:49
-----

Account: 0.0.0.1 /account 13640 0
Resource Name  Original Amount      New Amount      Difference
US Dollar      200.000000            20.000000      -180.000000
Total Resources 200.000000            20.000000      -180.000000
-----

+++++
The Total Rerate Impact:
+++++
Resource Name  Original Amount      New Amount      Difference
US Dollar      200.000000            20.000000      -180.000000
Total Resources 200.000000            20.000000      -180.000000
-----

END OF REPORT

```

=====
 The following is the detailed report for the example above.

PIN_RERATE DETAILED REPORT

=====
 Date Range: 3/1/2007 10:0:0 to 3/2/2007 20:4:49

Event: 0.0.0.1 /event/billing/product/fee/cycle/cycle_forward_monthly
 219550481834327128 0
 Service Type: /service/ip
 Event Time: 3/1/2007 20:4:14

O/N	Resource	Amount	Disct	GL_ID	Tax
Original	US Dollar	-200.000000	0.000	102	0.000000
New	US Dollar	20.000000	0.000	102	0.000000

Account: 0.0.0.1 /account 13640 0

Resource Name	Original Amount	New Amount	Difference
US Dollar	200.000000	20.000000	-180.000000
Total Resources	200.000000	20.000000	-180.000000

+++++

The Total Rerate Impact:

+++++

Resource Name	Original Amount	New Amount	Difference
US Dollar	200.000000	20.000000	-180.000000
Total Resources	200.000000	20.000000	-180.000000

END OF REPORT

Report Generated When Rerating Is Performed after Billing

The following example of a summary report shows rerating as a result of changing the monthly cycle forward fee from \$200 to \$20. Rerating took place *after* billing was run, so the difference is posted in the *next* bill.

Adjustments are posted only for cycle forward fees. Since cycle forward fees are charged in advance and since the difference between the old and the new amount in this case is \$180, the new amount is calculated as -360. In this case, the reports show only the adjusted amount.

PIN_RERATE SUMMARY REPORT

=====
 Date Range: 8/7/2007 10:0:0 to 9/7/2007 20:6:52

Resource Name	Original Amount	New Amount	Difference
US Dollar	0.000000	-360.000000	-360.000000
Total Resources	0.000000	-360.000000	-360.000000

+++++

The Total Rerate Impact:

+++++

Resource Name	Original Amount	New Amount	Difference
US Dollar	0.000000	-360.000000	-360.000000

Total Resources 0.000000 -360.000000 -360.000000

 END OF REPORT

The following is a detailed report of the example above:

PIN_RERATE DETAILED REPORT

=====

Date Range: 8/7/2007 10:0:0 to 9/7/2007 20:6:52

Event:0.0.0.1 /event/billing/adjustment/event 222875404996720238 0
 Adjusted From:0.0.0.1 /event/billing/product/fee/cycle/cycle_forward_monthly
 222330047229342470 0

Service Type: /service/ip
 Event Time: 9/7/2007 20:6:54

Resource	Amount	Disct	GL_ID
US Dollar	-180.000000	0.000	102

Event:0.0.0.1 /event/billing/adjustment/event 222875404996721006 0
 Adjusted From: 0.0.0.1 /event/billing/product/fee/cycle/cycle_forward_monthly
 222875404996719270 0

Service Type: /service/ip
 Event Time: 9/7/2007 20:6:54

Resource	Amount	Disct	GL_ID
US Dollar	-180.000000	0.000	102

Account: 0.0.0.1 /account 14854 0

Resource Name	Original Amount	New Amount	Difference
US Dollar	0.000000	-360.000000	-360.000000
Total Resources	0.000000	-360.000000	-360.000000

+++++

The Total Rerate Impact:

+++++

Resource Name	Original Amount	New Amount	Difference
US Dollar	0.000000	-360.000000	-360.000000
Total Resources	0.000000	-360.000000	-360.000000

 END OF REPORT

Improving pin_rerate Performance

The following parameters can be used with **pin_rerate** to improve performance:

- pin_rerate rerate_children 5
- pin_rerate rerate_per_step 1000
- pin_rerate rerate_fetch_size 5000

Part V

Pricing and Rerating Utilities

Part V includes utility reference information for pricing and rerating.

Part V contains the following chapters:

- [Pricing Utilities](#)
- [Rerating Utilities](#)

Pricing Utilities

This chapter provides reference information for Oracle Communications Billing and Revenue Management (BRM) pricing utilities.

load_brm_pricing

Use this utility to load real-time and pipeline pricing data into the database. This utility runs the following utilities to load the pricing data:

- **loadpricelist** loads the real-time pricing data into the BRM database.
- **LoadIfwConfig** loads the pipeline pricing data into the Pipeline Manager database.

For more information, see the following topics:

- [loadpricelist](#)
- "LoadIfwConfig" in *BRM Configuring Pipeline Rating and Discounting*

Location

BRM_Home/bin

Syntax

Load data:

```
load_brm_pricing [-v] [-d] [-f] -c Real_time_data_filename -F Pipeline_data_filename
```

Retrieve data:

```
load_brm_pricing [-v] [-d] [-f] -r Real_time_data_filename -F Pipeline_data_filename [-P] [-D] [-S] [-s ServiceType] [-t ModifiedTime] [-N name]
```

Get help:

```
load_brm_pricing -h
```

Parameters

-v

Displays information about successful or failed processing as the utility runs.

Note: This parameter is always used with other parameters and commands.

-d

Creates a log file for debugging purposes. Use this parameter for troubleshooting when the utility runs with no errors but the event map is not loaded into the database.

-f

Executes the command without prompting for a confirmation.

-c *Real_time_data_filename*

Reads the pricing data from *Real_time_data_filename* and commits it to the BRM database.

Important: When you use the **-c** parameter, it must immediately precede the file name. The **-v**, **-d**, and **-f** parameters can be used in any order, but must precede **-c Real_time_data_filename**.

-F Pipeline_data_filename

Reads the pricing data from or writes the pricing data to *Pipeline_data_filename*.

- When used with the **-c** parameter, **-F Pipeline data filename** reads the pricing data from *Pipeline data filename* and commits it to the Pipeline Manager database.
- When used with the **-r** parameter, **-F Pipeline data filename** retrieves the pricing data from the Pipeline Manager database and writes it to *Pipeline data filename*.

-r Real_time_data_filename [-P] [-D] [-S] [-s ServiceType] [-t ModifiedTime] [-N name]
Retrieves pricing objects from the BRM database and writes the pricing data to *Real_time_data_filename*.

Important: When you use the **-r** parameter, it must immediately precede the file name. The **-v**, **-d**, and **-f** parameters can be used in any order, but must precede **-r** and the file name. The parameters for determining the subset of information can be used in any order *after* the file name.

Use **-r Real_time_data_filename** with no additional parameters to retrieve all types of pricing objects from the BRM database.

Use the following parameters with **-r Real_time_data_filename** to retrieve only a subset of pricing objects from the BRM database:

- **-P** retrieves only the **/product** objects from the database.
- **-D** retrieves only the **/discount** objects from the database.
- **-S** retrieves only the **/sponsorship** objects from the database.
- **-s ServiceType** retrieves objects based on the specified service type. You can list multiple service types by using a comma (,) as a delimiter. For example: **-s /service/telco/gsm/telephony, /service/telco/gsm/data**.

Note: Do not use **-s ServiceType** with the **-S** parameter.

- **-t ModifiedTime** retrieves objects that were modified after the specified timestamp. You specify time using the ISO-8601 standard. [Table 31-1](#) lists the supported formats.

Table 31-1 Supported Formats

Format	Time Zone
YYYY	Local time of system used to run load_brm_pricing .
YYYY-MM	Local time of system used to run load_brm_pricing .
YYYY-MM-DD	Local time of system used to run load_brm_pricing .

Table 31-1 (Cont.) Supported Formats

Format	Time Zone
YYYY-MM-DDThh:mmZ	UTC
YYYY-MM-DDThh:mm:ssZ	UTC
YYYY-MM-DDThh:mm[+ -]TZD	TZD is the time zone relative to UTC. You can use a negative or positive offset from 00:00 to 12:00 (for example, -05:00 or +10:00).
YYYY-MM-DDThh:mm:ss[+ -]TZD	TZD is the time zone relative to UTC. You can use a negative or positive offset from 00:00 to 12:00 (for example, -05:00 or +10:00).

- **-N** *name* retrieves objects based on the specified product, discount, or sponsorship.

Note: You can use an asterisk (*) as a wildcard character to match zero or more characters in the name (for example, **-N *GSM***). Use **-N** with **-P**, **-D**, or **-S** to indicate the object type to be retrieved.

-h

Displays the syntax for the **load_brm_pricing** utility.

Results

The **load_brm_pricing** utility notifies you when it successfully completes a command.

If the utility does not notify you that it was successful, look in the utility log file to find any errors. The log file is either in the directory from which the utility was started or in a directory specified in the utility configuration file.

If an error occurs, enter the command again. The **load_brm_pricing** utility does not commit the pricing data to the database until it completes the command.

loadchangesets

Use this utility to import or export pipeline pricing data change sets. You can use the utility in interactive mode or non-interactive mode. In interactive mode, you enter single-word commands to perform individual actions.

You specify the BRM server to export from and the server to import to in a configuration file.

For more information, see "Exporting and importing change sets by using the loadchangesets utility" in *BRM Configuring Pipeline Rating and Discounting*.

Location

Pricing_Center_Home/lib

Syntax: Non-Interactive Mode

```
loadchangesets  [-v] [-fi change_set_file]
                [-v] [-fx change_set_file]
```

Parameters: Non-Interactive Mode

-v

Displays information about successful or failed processing as the utility runs.

-f change_set_file

Mandatory parameter that must precede **-i** and **-x**. Forces execution without prompting for confirmation.

-i change_set_file

When used with the **-f** parameter, imports change sets from *change_set_file* to the database. The file name must include the extension **.exp**.

When used alone, switches the utility to interactive mode. See "[Syntax: Interactive Mode](#)".

-x

When used with the **-f** parameter, exports the change sets from the database to *change_set_file*. Must be preceded with the **-f** parameter. The file name must include the extension **.exp**.

Syntax: Interactive Mode

```
loadchangesets -i

[read change_set_file]
[export]
[write change_set_file]
[import]
[help]
[quit]
[list]
```

Parameters: Interactive Mode

-i

Switches the utility to interactive mode. After entering interactive mode, commands are single words without the utility name.

export

Exports change sets from the database into memory.

write *change_set_file*

Writes the change sets stored in memory to *change_set_file*. Be sure to include the file name extension **.exp**.

read *change_set_file*

Reads change sets from *change_set_file* and stores them in memory. The file name must include the extension **.exp**.

import

Imports the change sets stored in memory to the database.

list

Lists the change sets currently stored in memory.

help

Displays help the syntax for the utility.

quit

Exits the utility.

Results: Export

Exported files are initially created in the *Pricing_Center_Home/export* directory. The status of the exported change sets changes to **Exported**. If all change sets are exported successfully, the file moves automatically into the */export/done* directory.

If there are any errors during export, the entire transaction is rolled back. In addition, the status of the change sets is automatically reset from **Exported** to **In Progress**.

Results: Import

The import process generates a log file called **import.log.0**. The log file contains information about validation errors, change sets that have passed validation, and change sets that have been imported successfully.

If all change sets are imported without any errors, the transaction is committed. If there are any errors, the entire transaction is rolled back.

load_event_map

Use this utility to load the event maps into the BRM database. You define the event maps in the *BRM_Home/sys/data/pricing/example/pin_event_map* file.

For more information about event maps, see "Mapping event types to services".

Caution: The **load_event_map** utility overwrites the entire event map. If you are updating the event map, you cannot load new mappings only. You load the entire event map each time you run the **load_event_map** utility. The **load_event_map** utility does not restrict to load any particular event type with a service. However, if the service types and event pairs are duplicated, the **load_event_map** utility reports an error.

Note: To connect to the BRM database, the **load_event_map** utility needs a configuration file in the directory from which you run the utility. See the discussion of configuration files in *BRM System Administrator's Guide*.

Location

BRM_Home/bin

Syntax

load_event_map [-d] [-v] *pin_event_map_file*

Parameters

-d

Logs messages to the **default.pinlog** file in the current directory, or in a directory specified in the utility configuration file. Use this parameter for troubleshooting when the utility runs with no errors, but the event map is not loaded into the database.

-v

Displays information about successful or failed processing as the utility runs.

Note: This parameter is always used with other parameters and commands. It is not position dependent. For example, you can enter **-v** at the beginning or end of a command to initiate the verbose parameter. To redirect the output to a log file, use the following syntax with the verbose parameter. Replace *filename.log* with the name of the log file:

load_event_map *other_parameter* **-v** > *filename.log*

pin_event_map_file

The name and location of the file that defines event maps. The default is *BRM_Home/sys/data/pricing/example/pin_event_map*.

If you copy *pin_event_map_file* to the same directory from which you run the **load_event_map** utility, you do not have to specify either the path or the file name.

If you run the command from a different directory from the one in which *pin_event_map_file* is located, you must include the entire path for the file.

Results

The **load_event_map** utility notifies you only if it encounters errors. Look in the default.pinlog file for errors. This file is either in the directory from which the utility was started, or in a directory specified in the utility configuration file.

load_pin_beid

Use this utility to load resources into the BRM database. You define the resources in the *BRM_Home/sys/data/pricing/example/pin_beid* file.

Tip: It is easier to use the Resource Editor to define resources.

For more information about defining resources, see "About resources".

Caution:

- The **load_pin_beid** utility overwrites the existing resources. If you are updating resources, you cannot load new resources only. You load complete sets of resources each time you run the **load_pin_beid** utility.
 - When you add or edit resources by using the Resource Editor, the Resource Editor does not update the **pin_beid** file. Therefore, if you want the **pin_beid** file to include all of your resources, you must edit it by hand.
-
-

Note: To connect to the BRM database, the **load_pin_beid** utility needs a configuration file in the directory from which you run the utility. See the discussion of configuration files in *BRM System Administrator's Guide*.

Location

BRM_Home/bin

Syntax

```
load_pin_beid [-d] [-v] pin_beid_file
```

Parameters

-d

Sends all flists used in creating the resources to a log file. Use this parameter for debugging when the utility appears to have run with no errors, but the resources do not appear in the database.

-v

Displays information about successful or failed processing as the utility runs.

Note: This parameter is always used with other parameters and commands. It is not position dependent. For example, you can enter **-v** at the beginning or end of a command to initiate the verbose parameter. To redirect the output to a log file, use the following syntax with the verbose parameter. Replace *filename.log* with the name of the log file:

load_pin_beid *other_parameter* **-v** > *filename.log*

pin_beid_file

The name and location of the file that defines resources. The default is *BRM_Home/sys/data/pricing/example/pin_beid*.

If you copy the **pin_beid** file to the same directory from which you run the **load_pin_beid** utility, you do not have to specify either the path or the file name.

Results

The **load_pin_beid** utility notifies you only if it encounters errors.

Important: You must restart the Connection Manager to make the new resources available.

load_pin_impact_category

Use this utility to load impact categories into the BRM database. You define the impact categories for the rate plan selector charges in the *BRM_Home/sys/data/config/pin_impact_category* file.

For information about impact categories, see "[Using Event Attributes to Rate Events in Real Time](#)".

Caution: The `load_pin_impact_category` overwrites the existing impact categories. If you are updating a set of impact categories, you cannot load new impact categories only. You load complete sets of impact categories each time you run the `load_pin_impact_category` utility.

Note: To connect to the BRM database, the `load_pin_impact_category` utility needs a configuration file in the directory from which you run the utility. See the discussion of configuration files in *BRM System Administrator's Guide*.

Location

BRM_Home/bin

Syntax

```
load_pin_impact_category pin_impact_category_file
```

Parameters

pin_impact_category_file

The name and location of the file that defines the impact categories. The default is *BRM_Home/sys/data/config/pin_impact_category*.

If you copy the `pin_impact_category` file to the same directory from which you run the `load_pin_impact_category` utility, you do not have to specify either the path or the file name.

Results

The `load_pin_impact_category` utility notifies you when it successfully creates the `/config/impact_category` storable object.

If the `load_pin_impact_category` utility does not notify you that it was successful, look in the `default.pinlog` file to find any errors. This file is either in the directory from which the utility was started or in a directory specified in the utility configuration file.

Important: You must restart the Connection Manager to make the new impact categories available.

load_pin_rum

You use this utility to load ratable usage metrics (RUMs) into the `/config/rum` storable object in the BRM database. You define RUMs in the `BRM_Home/sys/data/pricing/example/pin_rum` file.

For information about RUMs, see "About setting up RUMs for real-time rating".

Caution: The `load_pin_rum` utility overwrites the existing RUMs. If you are updating a set of RUMs, you cannot load new RUMs only. You load complete sets of RUMs each time you run the `load_pin_rum` utility.

Note: To connect to the BRM database, the `load_pin_rum` utility needs a configuration file in the directory from which you run the utility. See the discussion of configuration files in *BRM System Administrator's Guide*.

Location

`BRM_Home/bin`

Syntax

```
load_pin_rum [-d] [-v] pin_rum_file
```

Parameters

-d

Creates a log file for debugging purposes. Use this parameter for debugging when the utility appears to have run with no errors, but the RUMs do not appear in the database.

-v

Displays information about successful or failed processing as the utility runs.

Note: This parameter is always used with other parameters and commands. It is not position dependent. For example, you can enter `-v` at the beginning or end of a command to initiate the verbose parameter. To redirect the output to a log file, use the following syntax with the verbose parameter. Replace `filename.log` with the name of the log file:

```
load_pin_rumother_parameter -v > filename.log
```

pin_rum_file

The name and location of the file that defines RUMs. The default is `BRM_Home/sys/data/pricing/example/pin_rum`.

If you copy the `pin_rum` file to the same directory from which you run the `load_pin_rum` utility, you do not have to specify either the path or the file name.

Results

The **load_pin_rum** utility notifies you only if it encounters errors.

Important: You must restart the Connection Manager to make the new RUMs available.

load_pin_spec_rates

Use this utility to set up charging for administrative events by loading the contents of the **pin_spec_rates** file into the BRM database.

For information about charging for administrative events, see "[About Charging for Custom Events and Attributes](#)".

Caution: 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.

Note: 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 the discussion of configuration files in *BRM System Administrator's Guide*.

Location

BRM_Home/bin

Syntax

load_pin_spec_rates *pin_spec_rates_file*

Parameters

pin_spec_rates_file

The name and location of the file that maps opcodes to event types to be rated. The default is *BRM_Home/sys/data/config/pin_spec_rates*.

If you copy the **pin_spec_rates** file to the same directory from which you run the **load_pin_spec_rates** utility, you do not have to specify either the path or the file name.

Results

If the **load_pin_spec_rates** utility does not notify you that it was successful, look in the **default.pinlog** file to find any errors. This file is either in the directory from which the utility was started or in a directory specified in the utility configuration file.

Important: You must restart the Connection Manager for the new administrative event charges to take effect.

load_pin_sub_bal_contributor

Use this utility to load the configuration for contributor-based sub-balances into the BRM database. You define the sub-balance configuration in the *BRM_Home/sys/data/pricing/example/pin_sub_bal_contributor* file.

For information about contributor-based sub-balances, see "About tracking resources in account sub-balances".

Note: You cannot load separate */config/sub_bal_contributor* objects for each brand. All brands use the same object.

Caution: 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.

Important:

- The resources specified in the *pin_sub_bal_contributor* file must exist in the BRM database before running the *load_pin_sub_bal_contributor* utility. This includes any aggregation counters used to track consumption of resources that are shared among several accounts. Therefore, you must first load the *pin_beid* file by running the *load_pin_beid* utility. See "[load_pin_beid](#)".
 - 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 the discussion of configuration files 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.

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* storable 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.

Important: You must restart the Connection Manager to make the new sub-balance configurations available.

loadpricelist

Use this utility to prepare and commit price list information, such as products, deals, plans, offer profiles, and plan lists, to your BRM database. You define the price list or offer profile configurations by using the **price_list.xsd** file in the *BRM_Home/PricingCenter* directory.

Note: The **loadpricelist** utility is the XML Pricing Interface.

Important:

- The **loadpricelist** utility does not support the price list text format used by releases before Infranet Release 6.0.
 - Do not use the **loadpricelist** utility to assign custom rateable usage metrics (RUMs) to fold events in any rate plans. Fold events cannot use custom RUMs; therefore, products configured with them are rated incorrectly.
 - To connect to the BRM database, the CLASSPATH should point to an **Infranet.properties** file with the correct login information for the **loadpricelist** utility. The default **Infranet.properties** file is in the **/common** directory. To put the **Infranet.properties** file in the local directory with the **loadpricelist** utility, add period-slash (**./**) to the beginning of the CLASSPATH.
-

For information, see ["Using the XML Pricing Interface to Create a Price List"](#).

Location

BRM_Home/bin

Syntax: Non-Interactive Mode

Copy:

```
loadpricelist [-v] [-d] [-f] -c PriceListFile
```

Purge:

```
loadpricelist [-v] [-d] [-f] -p
```

Retrieve:

```
loadpricelist [-v] [-d] [-f] -r PriceListFile [-P] [-D] [-S] [-s ServiceType] [-t ModifiedTime]
```

Get help:

```
loadpricelist -h
```

Parameters: Non-Interactive Mode

-v

Displays information about successful or failed processing as the utility runs.

Note: This parameter is always used with other parameters and commands.

-d

Creates a log file for debugging purposes. Use this parameter for debugging when the utility appears to have run with no errors but the data has not been loaded into the database.

-f

Executes the command without prompting for a confirmation.

-c *PriceListFile*

Reads the price list information from *PriceListFile* and commits it to the BRM database.

Important:

When you use the **-c** parameter, it must immediately precede the file name. The **-v**, **-d**, and **-f** parameters can be used in any order, but must precede **-c** and the file name.

PriceListFile must be in the same directory as the **price_list.xsd** file. If they are not in the same directory, edit the **noNamespaceSchemaLocation** entry in *PriceListFile* to include the full path to **price_list.xsd**. By default, **price_list.xsd** is in the *BRM_Home*\PricingCenter directory. For example, if the XSD file is located in C:\pricelists, change the entry to:

```
xsi:noNamespaceSchemaLocation="C:\pricelists\price_list.xsd"
```

-p

Purges pricing objects from the BRM database.

-r *PriceListFile* [-P] [-D] [-S] [-s *ServiceType*] [-t *ModifiedTime*]

Retrieves pricing objects from the BRM database and writes them to *PriceListFile*.

When the **infranet.batchsize** entry is present in the **loadpricelist Infranet.properties** file, the utility retrieves objects in batches based on the value of the entry. See ["Downloading Price Lists in Batches"](#).

Use **-r *PriceListFile*** with no additional parameters to retrieve all types of pricing objects from the BRM database.

Use the following parameters with **-r *PriceListFile*** to retrieve only a subset of pricing objects from the BRM database:

- **-P** retrieves only the **/product** objects from the database.
- **-D** retrieves only the **/discount** objects from the database.
- **-S** retrieves only the **/sponsorship** objects from the database.
- **-s *ServiceType*** retrieves objects based on the specified service type. You can list multiple service types by using a comma (,) as a delimiter. For example: **-s /service/telco/gsm/telephony, /service/telco/gsm/data**.

Note: Do not use **-s *ServiceType*** with the **-S** parameter.

- **-t *ModifiedTime*** retrieves objects that were modified after the specified timestamp. You specify time using the ISO-8601 standard. [Table 31-2](#) lists the supported formats:

Table 31–2 Supported Formats

Format	Time Zone
YYYY	Local time of system used to run loadpricelist .
YYYY-MM	Local time of system used to run loadpricelist .
YYYY-MM-DD	Local time of system used to run loadpricelist .
YYYY-MM-DDThh:mmZ	UTC
YYYY-MM-DDThh:mm:ssZ	UTC
YYYY-MM-DDThh:mm[+ -]TZD	TZD is the time zone relative to UTC. You can use a negative or positive offset from 00:00 to 12:00 (for example, -05:00 or +10:00).
YYYY-MM-DDThh:mm:ss[+ -]TZD	TZD is the time zone relative to UTC. You can use a negative or positive offset from 00:00 to 12:00 (for example, -05:00 or +10:00).

Important: When you use the **-r** parameter, it must immediately precede the file name. The **-v**, **-d**, and **-f** parameters can be used in any order, but must precede **-r** and the file name. The parameters for determining the subset of information can be used in any order *after* the file name.

-h

Displays the syntax for the **loadpricelist** utility.

Syntax: Interactive Mode

```
loadpricelist  read PriceListFile

               write
                 PriceListFile [planlist [type] | plan |
                               | bestpricing | deal | product | discount |
                               | sponsorship | offer_profile] [PriceObjectName]

               write
                 PriceListFile [dependency | transition]

               delete
                 [planlist [type] | plan | bestpricing | deal |
                 | product | discount | sponsorship |
                 offer_profile] [PriceObjectName]

               delete
                 [dependency | transition]

               retrieve
                 [-s ServiceType] [-t ModifiedTime] [product]
                 [discount] [sponsorship]

               commit
               purge
               help
               quit
               verbose [on | off | debug]
               list
```

Parameters: Interactive Mode

read *PriceListFile*

Reads the price list information from *PriceListFile* and stores it in internal memory as an flist.

Caution: `loadpricelist` maintains only one flist. When you load the price list information from your database, `loadpricelist` removes the current flist and creates a new flist.

Important: *PriceListFile* must be in the same directory as the `price_list.xsd` file. If they are not in the same directory, edit the `noNamespaceSchemaLocation` entry in *PriceListFile* to include the full path to `price_list.xsd`. By default, `price_list.xsd` is in the `BRM_Home\PricingCenter` directory. For example, if the XSD file is located in `C:\pricelists`, change the entry to:
`xsi:noNamespaceSchemaLocation="C:\pricelists\price_list.xsd"`

write

Writes the price list information from the in-memory internal flist into *PriceListFile*.

```
write PriceListFile [planlist [type] | plan | bestpricing | deal | product |
discount | sponsorship | offer_profile] [PriceObjectName]
```

```
write PriceListFile [dependency | transition]
```

Use `write PriceListFile` without any other parameters to write *all* pricing objects from the in-memory internal flist into *PriceListFile*.

Use the following parameters to write only a subset of pricing objects from the in-memory internal flist into *PriceListFile*:

- `planlist [type]` specifies to write only plan list information into *PriceListFile*. You can optionally specify to write only a plan list object of a particular type into *PriceListFile*.
- `plan` specifies to write only the `/plan` objects into *PriceListFile*.
- `bestpricing` specifies to write only the `/bestpricing` objects into *PriceListFile*.
- `deal` specifies to write only the `/deal` objects into *PriceListFile*.
- `product` specifies to write only the `/product` objects into *PriceListFile*.
- `discount` specifies to write only the `/discount` objects into *PriceListFile*.
- `sponsorship` specifies to write only the `/sponsorship` objects into *PriceListFile*.
- `offer_profile` specifies to write only the `/offer_profile` objects into *PriceListFile*.
- `PriceObject` specifies to write only the specified pricing object into *PriceListFile*.
- `dependency` specifies to write only `/dependency` objects into *PriceListFile*.
- `transition` specifies to write only `/transition information` objects into *PriceListFile*.

delete

Deletes in-memory price list information from the BRM database.

```
delete [planlist [type] | plan | bestpricing | deal |
```

```

    | product | discount | sponsorship | offer_profile]
  [PriceObject]

```

```
delete [dependency | transition]
```

Use **delete** without any other parameters to delete *all* in-memory pricing information from the database.

Use the following parameters to delete only a subset of in-memory pricing information from the database:

- **planlist** [*type*] specifies to delete the plan list information of a particular type.
- **plan** specifies to delete only the **/plan** objects from the database.
- **bestpricing** specifies to delete only the **/bestpricing** objects from the database.
- **deal** specifies to delete only the **/deal** objects from the database.
- **product** specifies to delete only the **/product** objects and all related product information, such as the rate plan, rate, rate tier, and balance impact, from the database.
- **discount** specifies to delete only the **/discount** objects from the database.
- **sponsorship** specifies to delete only the **/sponsorship** objects from the database.
- **offer_profile** specifies to delete only the **/offer_profile** objects from the database.
- *PriceObject* specifies to delete only the specified pricing object from the database.
- **dependency** specifies to delete only **/dependency** objects.
- **transition** specifies to delete only **/transition** objects.

retrieve

Retrieves the specified price list objects from the BRM database and stores it in internal memory as an flist.

```
retrieve    [-s ServiceType] [-t ModifiedTime] [product]
              [discount] [sponsorship]
```

Use **retrieve** without any other parameters to retrieve *all* pricing objects from the BRM database.

Use the following parameters to retrieve only a subset of pricing objects from the BRM database:

- **-s** *ServiceType* retrieves objects based on the specified service type. Multiple service types are delimited with a comma (,).

Do not use this parameter with the **sponsorship** parameter.

- **-t** *ModifiedTime* retrieves objects that were modified after the specified time. You specify time using the ISO-8601 standard. [Table 31–3](#) lists the supported formats:

Table 31–3 Supported Formats

Format	Time Zone
YYYY	Local time of system used to run loadpricelist .
YYYY-MM	Local time of system used to run loadpricelist .
YYYY-MM-DD	Local time of system used to run loadpricelist .
YYYY-MM-DDThh:mmZ	UTC

Table 31-3 (Cont.) Supported Formats

Format	Time Zone
YYYY-MM-DDThh:mm:ssZ	UTC
YYYY-MM-DDThh:mm[+ -]TZD	TZD is the time zone relative to UTC. You can use a negative or positive offset from 00:00 to 12:00 (for example, -05:00 or +10:00).
YYYY-MM-DDThh:mm:ss[+ -]TZD	TZD is the time zone relative to UTC. You can use a negative or positive offset from 00:00 to 12:00 (for example, -05:00 or +10:00).

- **product** retrieves only the **/product** objects from the database.
- **discount** retrieves only the **/discount** objects from the database.
- **sponsorship** retrieves only the **/sponsorship** objects from the database.

commit

Commits the pricing information to the database.

purge

Removes all price objects from the database.

help

Displays the syntax for the **loadpricelist** utility.

quit

Exits the utility.

verbose on | off | debug

Sets verbose information.

- **verbose on** displays the status of the command most recently executed.
- **verbose off** displays the status only if there is an error.
- **verbose debug** logs the status of the command to the log file specified in the **Infranet.properties** file. The default log file is **javapcm.log**.

list

Prints to the screen all price list information currently stored in internal memory. Because the in-memory price list is in flist format, the utility displays the price list in flist format.

Results

The **loadpricelist** utility notifies you when it successfully completes a command.

If the utility does not notify you that it was successful, look in the **javapcm.log** file to find any errors. This log file is either in the directory from which the utility was started or in a directory specified in the Java property file.

If an error occurs, enter the command again. The **loadpricelist** utility does not save changes until it completes the command.

load_usage_map

Use this utility to load the usage map into the **config/usage_map/system** storable object in the BRM database. You define the usage map in the *BRM_Home/sys/data/pricing/example/pin_usage_map* file.

For more information, see "Mapping event types to RUMs".

Caution: The **load_usage_map** overwrites the existing usage maps. If you are updating a set of usage maps, you cannot load new usage maps only. You load complete sets of usage maps each time you run the **load_usage_map** utility.

Note: To connect to the BRM database, the **load_usage_map** utility needs a configuration file in the directory from which you run the utility. See the discussion of configuration files in *BRM System Administrator's Guide*.

Location

BRM_Home/bin

Syntax

```
load_usage_map [-d] [-v] pin_usage_map_file
```

Parameters

-d

Creates a log file for debugging purposes. Use this parameter for debugging when the utility appears to have run with no errors, but the usage maps do not appear in the database.

-v

Displays information about successful or failed processing as the utility runs.

Note: This parameter is always used with other parameters and commands. It is not position dependent. For example, you can enter **-v** at the beginning or end of a command to initiate the verbose parameter. To redirect the output to a log file, use the following syntax with the verbose parameter. Replace *filename.log* with the name of the log file:

```
load_usage_map other_parameter -v > filename.log
```

pin_usage_map_file

The name and location of the file that defines usage maps. The default is *BRM_Home/sys/data/pricing/example/pin_usage_map*.

If you copy the **pin_usage_map** file to the same directory from which you run the **load_usage_map** utility, you do not have to specify either the path or the file name.

Results

The **load_usage_map** utility notifies you only if it encounters errors.

Universal Event Loader

Use the Universal Event (UE) Loader to load event records from event log files into the BRM database. For information about loading events, see ["About Rating Events Created by External Sources"](#) and ["Loading Events from External Sources"](#).

Note: To connect to the database, the UE Loader needs a configuration file in the directory from which you run the utility. See the discussion of configuration files in *BRM System Administrator's Guide*.

UE Loader runs in three modes:

- **Preprocessing mode (-m parse).** In the preprocessing mode, UE Loader parses the event log file and writes data to an intermediate file, including which accounts to load events for.
- **Event loading mode (-m load).** In event loading mode, UE Loader uses the data in the intermediate file (created in preprocessing mode) to load events into the BRM database.
- **Combined mode.** Use combined mode to run both preprocessing and event loading modes without stopping. This is the default mode.

UE Loader records all errors that occur in these modes in the log file *BRM_Home/apps/uel/event_log_file_name_lerr.xml*. It also records all event load successes in the log file *BRM_Home/apps/uel/event_log_file_name_lsucc.xml*.

For details, see ["Troubleshooting Event Loading"](#).

Location

BRM_Home/apps/uel

Syntax

```
uel -t template_name [-v] [-m parse | load] [-test] event_log_file_name
```

Parameters

-t *template_name*

The name of the event import template.

-v

Sends error messages or debug messages to the console according to the debug level you set in the **Infranet.properties** file.

-m *parse* | *load*

The mode of operation. You specify the mode of operation only when testing the import template. When you load events in a production database, you do not need to specify the mode of operation.

- Use **parse** to read the event log file and store the results in an intermediate cache file instead of in the BRM database.

- Use **load** to load the results contained in the intermediate cache file that was created by using the **parse** entry. The UE Loader loads the events into the BRM database.
- To parse the event log file and load events in one operation, do not enter the **-m** parameter. This is the default mode in a production database.

-test

Runs the program, but loads events in calculate-only mode. To test how events are loaded, read the output flist.

event_log_file_name

The file name of the event log file from which you are loading events.

Note: Do not enter the file path of the event log file. You specify the file path in the properties file. See "[Specifying the Event Log File Location](#)".

Results

For information about troubleshooting UE Loader, see "[Troubleshooting Event Loading](#)".

Rerating Utilities

This chapter provides reference information for Oracle Communications Billing and Revenue Management (BRM) rerating utilities.

load_pin_config_ood_criteria

Use the `load_pin_config_ood_criteria` utility to load out-of-order criteria into the `/config/event_order_criteria` object in the BRM database. You define out-of-order criteria in the `pin_config_ood_criteria.xml` file in `BRM_Home/sys/data/config`.

For more information, see "[Defining Out-of-Order Criteria](#)" and "[Loading Out-of-Order Criteria](#)".

Important: To connect to the BRM database, the `load_pin_config_ood_criteria` utility needs a configuration (`pin.conf`) file in the directory from which you run the utility.

Caution: The `load_pin_config_ood_criteria` utility overwrites existing out-of-order criteria. If you are updating out-of-order criteria, you cannot load new out-of-order criteria only. You must load complete sets of out-of-order criteria each time you run the `load_pin_config_ood_criteria` utility.

Location

`BRM_Home/bin`

Syntax

```
load_pin_config_ood_criteria [[-d] [-v] | [-t] out_of_order_criteria_file.xml ] | [-h]
```

Parameters

-h

Displays online help about the command.

-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 detailed information as the utility runs.

-t

Checks the validity of the XML file but does not process any data.

out_of_order_criteria_file

The name and location of the file that defines out-of-order criteria. The default `pin_config_ood_criteria.xml` file is in `BRM_Home/sys/data/config`.

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_config_ood_criteria BRM_Home/sys/data/config/pin_config_ood_criteria.xml
```

Tip: If you copy the **pin_config_ood_criteria.xml** file to the directory from which you run the **load_pin_config_ood_criteria** utility, you do not have to specify the path or file name. By default, the name of the file is **pin_config_ood_criteria.xml**. You can change this name.

Results

The **load_pin_config_ood_criteria** utility notifies you when it successfully creates the **/config/event_order_criteria** object. Otherwise, look in the **default.pinlog** file for errors. This file is either in the directory from which the utility was started or in a directory specified in the utility configuration file.

To verify that out-of-order criteria was loaded, you can display the **/config/event_order_criteria** object by using the Object Browser application in Developer Center, or use the **robj** command with the **testnap** utility.

Important: You must send a **Reload** semaphore to make new out-of-order criteria available in the Pipeline Manager. The **Reload** semaphore is used by the **DAT_PortalConfig** data module.

load_pin_rerate_flds

Use this utility to load the following information into the BRM database:

- Load extraction keys-to-event field mappings into the `/config/rerate_flds` object. You define extraction keys and map them to EDR fields by using XML. For more information, see ["Defining Custom pin_rerate Parameters for Rerating"](#).
- Load balance-impact comparison fields into the `/config/rerate_flds/compare_bi` object. The balance impact fields in this object determine whether the balance impacts of rerating and previous rating are equivalent. For more information, see ["Determining Whether Balance Impacts of Rerating and Previous Rating Are Equivalent"](#).

Location

`BRM_Home1 sys/data/config`

Run `load_pin_rerate_flds` from this directory.

Syntax

```
load_pin_rerate_flds -f xml_file_name [-v] [-h]
```

Parameters

-f *xml_file_name*

Specifies the name of the XML file that contains either the extraction keys and field mappings or the balance-impact comparison fields.

Important: The file you load cannot include both extraction-key mapping and balance-impact comparison fields. You must load the files for these configurations separately.

-verbose

Displays information about successful or failed processing as the utility runs.

Note: This parameter is always used with other parameters and commands. It is not position dependent. For example, you can enter **-verbose** 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_rerate_flds any_other_parameter -v > filename.log
```

-help

Displays the syntax and parameters for this utility.

Results

This utility notifies you when it successfully creates the `/config/rerate_flds` object.

If it cannot create the **/config/rerate_flds**, it logs an error in the log file (**default.pinlog**). It creates the log file either in the directory from which the utility was started or in the directory specified in the configuration file.

pin_event_extract

Use this utility to extract events from your database to be rerated by the BRM Pipeline Manager.

You define which events to extract by using the **pin_event_extract.cfg** file located in the *BRM_Home/apps/pin_event_extract* directory. For information, see ["Specifying Which Events to Extract for Rerating"](#).

For information about extracting events, see ["Running the pin_event_extract Utility"](#).

Note: To connect to the BRM database, the **pin_event_extract** utility needs a configuration file in the directory from which you run the utility.

Location

BRM_Home/bin

Syntax

```
pin_event_extract [-f ConfigFileName] [-r roaming_input_file] [-b BrandName] [-e] [-o] [-h]
```

Parameters

-f *ConfigFileName*

Specifies the name and location of the configuration file that defines which events to extract. For information on how to create the file, look at the sample file (*BRM_Home/apps/pin_event_extract/pin_event_extract.cfg*).

By default, the utility looks in the current working directory. If your configuration file is located in a different directory, you must specify the entire path for the file. If the **pin_event_extract.cfg** file is located in the current working directory, you can ignore this parameter.

-r *roaming_input_file*

Extracts events based on the information provided in the *roaming_input_file*. The *roaming_input_file* is generated by the RAP Processing pipeline during Outcollect processing. It contains records that specify the extraction criteria for extracting TAP settlement records from the BRM database. For more information, see ["Backing Out the Balance Impacts for the Rejected TAP Files and Records"](#) in *BRM Configuring Roaming in Pipeline Manager*.

-b *BrandName*

Extracts events assigned to a specific brand. You can use this option alone or with the **-f** option.

-e

Flags the events for back-out-only rerating. Pipeline Manager backs out the balance impact and does not rerate the event.

-o TRUE | FALSE

Overrides the program lock.

-o TRUE resets the status table to the unused state.

-o FALSE sets the status table to the used state.

The Event Extraction Manager cannot run at the same time as Rated Event Loader or Rerated Event Loader. To prevent this, all three applications use a status table to determine if one of the other applications is running. If one of the applications terminates abnormally and leaves the status table in a locked state, the Event Extraction Manager cannot be started. In this case, use the **-o TRUE** option to reset the status table to an unused state.

Important: Before using this option, ensure that the Rated Event Loader and Rerated Event Loader are stopped.

-h

Displays the syntax and parameters for this utility.

Results

The **pin_event_extract** utility notifies you when it successfully completes a command.

pin_load_rerate_jobs

This utility is used by the **OODHandler** batch handler to process out-of-order events. It converts rerate-request files generated by the FCT_EventOrder module into flist format and then calls PCM_OP_ACT_USAGE to generate a notification event.

For more information, see "[About Automatic Rerating of Out-of-Order Events](#)".

Important: You must run the utility from a directory that contains both the **pin_rerate_job_info.xsd** file and a **pin.conf** configuration file. By default, these files are located in the *BRM_Home/apps/pin_ood_handler/process* directory.

Location

BRM_Home/bin

Syntax

```
pin_load_rerate_jobs [[-d] [-v] | [-t] rerate_request_file.xml] | [-h]
```

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 detailed information as the utility runs.

-t

Checks the validity of the XML file but does not process any data. When you run the file in test mode, it returns either that the XML file is valid or an error message that contains the invalid line number.

rerate_request_file.xml

Specifies the name of the rerate request XML file that is generated by the FCT_EventOrder pipeline module.

-h

Displays online help about the command.

Results

The utility notifies you only if it encounters errors. Look in the **default.pinlog** file for errors. This file is either in the directory from which the utility was started or in a directory specified in the utility configuration file.

pin_rate_change

Use this BRM utility after you change the rates for cycle forward and cycle forward arrears events in the current cycle. This utility triggers the creation of rerating requests that the **pin_rerate** utility uses to recalculate the charges for these events and adjust the balance impacts.

For information on rate changes in a cycle, see "About Calculating Charges When You Change the Rate" in *BRM Configuring and Running Billing*.

For information on the **pin_rerate** utility, see "[pin_rerate](#)".

The **pin.conf** file for this utility is in *BRM_Home/apps/pin_rate_change*.

Location

BRM_Home/bin

Syntax

```
pin_rate_change [-v] [-d] [-h]
```

Parameters

-v

Displays information about successful or failed processing as the utility runs.

Note: This parameter 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_rate_change any_other_parameter -v > filename.log
```

-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.

-h

Displays the syntax and parameters for this utility.

Results

The **pin_rate_change** utility notifies you when it successfully creates the **rerating requests**.

If the **pin_rate_change** utility does not notify you that it was successful, look in the utility log file (**default.pinlog**) to find any errors. The log file is either in the directory from which the utility was started, or in a directory specified in the configuration file.

pin_rel

Loads batches of prepared or related event records from Pipeline Manager into the BRM database.

There are two ways to use this utility. When you initially run **pin_rel**, use the command without any options and use the file name as the only command-line parameter. You use the **override** option when the utility has not successfully completed its process and needs to be rerun.

pin_rel records any errors in the **pin_rel** log file (*BRM_Home/apps/pin_rel/rel.pinlog*).

Location

BRM_Home/apps/pin_rel

Syntax

```
pin_rel [-override] event_file_name
```

Parameters

-override

This option starts a new **pin_rel** process. Use this option to restart **pin_rel** when it has abnormally stopped or you have stopped it manually.

Important: Use this option only when you know there are no other RE Loader processes running.

RE Loader maintains a status of its operations. Because only one RE Loader process can load events into the same database at the same time, the status must indicate the loading process is complete before another process can start loading. If you manually stop **pin_rel**, its status may not reflect its true state. The **-override** parameter overrides the status and permits a new loading process to start, providing one is not already running.

event_file_name

The name of the event file to load, including its extension.

Results

If **pin_rel** is successful, it returns PROCESSED SUCCESSFULLY in the RE Loader log file (*BRM_Home/apps/pin_rel/rel.pinlog*).

If an error occurs during loading, this utility aborts the loading process. An error is logged in the **rel.pinlog** file, SQL loader errors are logged in a "bad" file (*BRM_Home/apps/pin_rel/CDR_file_name.bad*), and the records loaded in this session are deleted from the database.

If an error occurs during account updating, the error is logged in the **rel.pinlog** file. Loaded records are not deleted from the database.

pin_rerate

Use this BRM utility to rerate events rated in real-time and events rated in batch by Pipeline Manager, and update account balances with the rerated results. For more information, see "[Using the pin_rerate Utility](#)".

If you rerate both real-time-rated and pipeline-batch-rated events concurrently, you run **pin_rerate** multiple times: once to create rerating jobs by selecting the accounts and events for rerating, and several times again to process the rerate jobs.

If you do not use Pipeline Manager for batch rating, you select the accounts and events and rerate them by using a single **pin_rerate** command.

Note:

- **pin_rerate** is a multithreaded application.
 - In a multischema environment, you must run **pin_rerate** separately on each database schema.
 - To connect to the BRM database, the **pin_rerate** utility needs a configuration file in the directory from which you run the utility.
-
-

Location

BRM_Home/bin

The **pin.conf** file for this utility is located in *BRM_Home/apps/pin_rerate*. Run **pin_rerate** from this directory.

Syntax

```
pin_rerate [-t [ss/mm/hh/] MM/DD/YYYY]
           [-a account POID_id]
           [[-d | -g | -i | -m | -n | -p | -s] input_file]
           [-line subscription_id]
           [-field_name input_file]
           [-r]
           [-reason reason_code]
           [-b [c | e]]
           [-c]
           [-backout]
           [-e [-a | -s | -p | -n | -d]]
           [-h | help]
           [-purge [-t [ss/mm/hh/] MM/DD/YYYY]]
           [-l [d | s | sd | n]]
           -process jobs [-reason reason_code [,reason_code...]]
           -process queue
           -rerate [-reason reason_code [,reason_code...]] [-l [d | s | sd | n]]
           -process recycle [-db database_id]
```

Parameter Overview

The parameters specify which accounts and events to rerate and which rerating process to perform.

- To select the accounts and events for rerating, see the following sections:
 - [Parameters for Selecting Accounts for Rerating](#)

- [Parameter for Specifying Events for Rerating](#)
- To assign a rerate reason code for rerating, see "[Parameter for Assigning a Rerate Reason](#)".
- To specify rerating behavior, such as how to order the events, whether to update the database, and whether to generate reports, see "[Parameters Affecting Rerating Behavior](#)".
- To purge rerate jobs, see "[Parameter for Purging Rerate Jobs](#)".
- To process rerate jobs and rerate the selected accounts and events, see "[Parameters for Processing Rerate Jobs](#)".

Parameters for Selecting Accounts for Rerating

The following parameters are used to select accounts for rerating based on the described event criteria.

All account selection parameters are optional and mutually exclusive, except for the **-t** parameter, which is mandatory when selecting the accounts and events to rerate. If you specify only the time (**-t**) parameter, BRM selects all accounts for rerating in the period defined by the **-t** parameter.

For more information about selecting account for rerating, see "[Specifying Accounts for Rerating](#)".

-a [account POID_id]

Selects a *single account* for rerating based on the provided account POID.

When you specify an account POID, you can also use the **-c** option to rerate events *without* updating the database.

This parameter is optional.

Note: You cannot use this parameter with the **-d, -g, -i, -m, -p, -s, -line**, or custom parameters.

-d input_file

Selects accounts for rerating based on the *deal*. Accounts that have events associated with the products and discounts that belong to the deals specified in *input_file* are selected for rerating.

input_file format: A text file with one deal name specified on each line. The deal names are case sensitive.

This parameter is optional.

Important: If a product is part of multiple deals, all accounts that purchase the product are selected for rerating even if they did not purchase the deal that was rerated.

Note: You cannot use this parameter with the **-a, -g, -i, -m, -n, -p, -s, -line**, or custom parameters.

-g *input_file*

Selects accounts for rerating based on one of the following:

- The account and bill unit objects
- The account, bill unit, and balance group objects

Accounts that match any criteria listed in *input_file* are selected for rerating.

- If items in *input_file* specify an account, bill unit, and balance group, only events specific to the balance group are selected.
- If items in *input_file* specify an account and a bill unit only, **pin_rerate** searches for all the balance groups associated with the bill unit and then selects the events specific to those balance groups.

input_file format: A text file containing information for one or more accounts in flist format. Specify an account POID, **/billinfo** POID, and balance group POID in a results array for each account. For example:

```
0 PIN_FLD_RESULTS                ARRAY [1]
  1 PIN_FLD_ACCOUNT_OBJ           POID [0] $DB_NO /account 12345 0
  1 PIN_FLD_BILLINFO_OBJ          POID [0] $DB_NO /billinfo 34567 0
  1 PIN_FLD_BAL_GRP_OBJ           POID [0] $DB_NO /balance_group 66765 0
0 PIN_FLD_RESULTS                ARRAY [1]
  1 PIN_FLD_ACCOUNT_OBJ           POID [0] $DB_NO /account 12344 0
  1 PIN_FLD_BILLINFO_OBJ          POID [0] $DB_NO /billinfo 45654 0
  1 PIN_FLD_BAL_GRP_OBJ           POID [0] $DB_NO /balance_group NULL 0
...
```

where:

- PIN_FLD_ACCOUNT_OBJ specifies the POID of the account object.
- PIN_FLD_BILLINFO_OBJ specifies the POID of the bill unit object.
- PIN_FLD_BAL_GRP_OBJ specifies the POID of the balance group object.

To rerate all events for all balance groups for a given account's bill unit, specify a value of NULL as the POID ID for the balance group object, as shown in the second results array in the above example.

This parameter is optional.

Note: You cannot use this parameter with the **-a, -d, -i, -m, -n, -p, -s, -e, -line**, or custom parameters.

-i *input_file*

Selects accounts for rerating based on the *discount object*. Accounts that own at least one instance of the discount objects specified in *input_file* are selected for rerating.

input_file format: A text file with one discount name specified on each line. Discount names are case sensitive.

This parameter is optional.

Note: You cannot use this parameter with the **-a, -d, -g, -m, -n, -p, -s, -line**, or custom parameters.

-m *input_file*

Selects a set of accounts for rerating based on the provided account POIDs.

input_file format: A text file containing the accounts to rerate in flist format. Specify each account in a results array. For example:

```
0 PIN_FLD_RESULTS                ARRAY [1]
  1 PIN_FLD_ACCOUNT_OBJ          POID [0] $DB_NO /account 12345 0
0 PIN_FLD_RESULTS                ARRAY [2]
  1 PIN_FLD_ACCOUNT_OBJ          POID [0] $DB_NO /account 12333 0
...
```

where PIN_FLD_ACCOUNT_OBJ is the POID of the account object.

Note: You cannot use this parameter with the **-a, -d, -g, -i, -n, -p, or -s -e, -line**, or custom parameters.

-n input_file

Selects accounts for rerating based on the *event types*. Accounts that have events of the types specified in *input_file* are selected for rerating.

input_file format: A text file with one event type specified on each line. Event types are case sensitive.

This parameter is optional.

Note:

You cannot use this parameter with the **-a, -d, -g, -i, -m, -p, -s, -line**, or custom parameters.

If you use this parameter with the **-r** parameter, all subclasses of the event type specified in the input file are also rerated. For more information, see "[Specifying Events for Rerating](#)".

When using a custom **pin_rerate** parameter, the **-n** parameter is mandatory if the custom parameter is based on an event field that is present only in an event subclass. In this case, the **-n** parameter input file can contain only one event type.

-p input_file

Selects accounts for rerating based on the *product*. Accounts that have events associated with the products specified in *input_file* are selected for rerating.

input_file format: A text file with one product name specified on each line. Product names are case sensitive.

This parameter is optional.

Note: You cannot use this parameter with the **-a, -d, -g, -i, -m, -n, -s, -line**, or custom parameters.

-s input_file

Selects accounts for rerating based on the *service type*. Accounts that have events associated with the service types specified in *input_file* are selected for rerating.

input_file format: A text file with one service type specified on each line. Service type names are case sensitive.

This parameter is optional.

Note: You cannot use this parameter with the **-a, -d, -g, -i, -m, -n, -p, -line**, or custom parameters.

-t [ss/mm/hh/]MM/DD/YYYY

Selects accounts for rerating based on the event *start time*. All accounts with events that occurred between the start time and the current date are selected for rerating.

Note: The group of selected accounts can be further restricted by using criteria specified with the **-a, -d, -g, -i, -m, -n, -p, -s, -line**, or custom parameters.

-line *subscription_id*

Selects accounts for rerating based on a subscription service.

subscription_id specifies the subscription service ID. The subscription service is identified by the PIN_FLD_ALIAS_LIST.PIN_FLD_NAME field, which can specify, for example, the caller ID such as a phone number or the MAC address.

This parameter is optional.

Note: The group of selected accounts can be further restricted by using criteria specified with the **-a, -d, -g, -i, -m, -n, -p, -s**, or custom parameters.

-field_name input_file

field_name is a custom **pin_rerate** parameter you have defined that maps to a specific event field.

input_file is a file that contains the values of the event field that are used to select the accounts for rerating.

Selects accounts based on the custom parameter. Accounts are selected for rerating that have events with the field identified by *field_name* whose field value is one of those specified in *input_file*.

For information about defining custom **pin_rerate** parameters, see ["Defining Custom pin_rerate Parameters for Rerating"](#).

Note: You cannot use this parameter with the **-a, -d, -g, -i, -m, -p, -s**, or **-line** parameters.

Important: If the custom parameter maps to a field in an **/event** subclass, you must specify the subclass event type by including the **-n** parameter, and the **-n** parameter input file can include only one event type. If you specify more than one event type, an error is returned.

Parameter for Specifying Events for Rerating

Use the following parameter to specify which events to rerate for the selected accounts.

For more information about selecting events for rerating, see ["Specifying Events for Rerating"](#).

-r

Specifies whether to use *selective rerating*. This applies the account selection criteria parameters (**-a**, **-d**, **-g**, **-i**, **-m**, **-n**, **-p**, **-s**, **-line** or `field_name`) to the account's events as well and selects events for rerating based on the specified parameter. For example, using **-r** and **-p** `input_file` in the command line selects accounts based on the products specified in `input_file`, and then selects and rerates only the account's events associated with those products.

Important:

Do not use selective rerating if your rating scheme includes credit limits or resource-based tiers. These schemes require that all events related to an account are rerated to ensure accurate rerating.

Do not use selective rerating if deferred taxation is used for taxing events during rerating.

Use caution when specifying selective rerating. Account balances can be impacted by different types of events, and the order of the balance impacts is important to accurately apply discounts and consume resources. It is typically safer to rerate all of an account's events.

This parameter is optional and can be used with any other parameter. The value for this parameter can be specified as either **0 (disable)** or **1 (enable)**.

Note: If no value is specified for this parameter, the default behavior is to consider the value as **1**.

Parameter for Assigning a Rerate Reason

-reason *reason_code*

Use with other selection parameters to assign a rerate reason code to the jobs created for the selected accounts.

For more information about assigning rerate reason codes, see ["Assigning Rerate Reasons to Rerate Jobs"](#).

Reason code format: Any integer except **1** (**1** is a reserved default value for pipeline-generated rerating requests).

If you create rerate jobs through BRM automatic rerating, rerate reasons are hard coded by the opcodes that handle those rerating scenarios. Be sure your rerate reasons are unique to process rerate jobs associated with them during separate rerating executions.

To process rerate jobs according to a rerate reason, see ["Parameters for Processing Rerate Jobs"](#).

Parameters Affecting Rerating Behavior

The following are additional, optional parameters that affect rerating behavior and the utility's output.

-b [c | e]

Specifies the order in which *batch* events are rated. The **c** option rerates events in the order that they were created in the BRM database. The **e** option rerates events based on the time when the event actually occurred. The default is **e**. For more information, see "[Specifying the Event Sequence for Rerating](#)".

Note: Due to real time account balance impacts and credit limit monitoring, the order in which the batch events are processed may result in different rating results. For more information, see "[About Rerating Pipeline and Real-Time Events Concurrently](#)".

This parameter is optional.

-c

Specifies calculation of rerating only. Use this parameter to log the rerating result of an account into a report file without updating the database. This option can be used for only one account, so it works only when using the **-a** parameter.

This parameter is optional.

-backout

Specifies back-out-only rerating, which backs out the balance impacts of rating without rerating events.

For more information about back-out-only rerating, see "[Using pin_rerate for Back-Out-Only Rerating](#)".

This parameter is optional and can be used with any other parameter.

Note: When choosing the events to back out, ensure that you do not select events that could imbalance your general ledger, such as events for which fees have already been paid and usage events that should be rerated. Typically, back-out-only rerating is performed only on usage events where rating should not have occurred.

-e [-a | -s | -p | -n | -d]

Returns an estimate of how many events might be affected by rerating based on which accounts are being rerated. Options:

-e -a: A single account

-e -s: Accounts with a specific service type

-e -p: Accounts with specific products

-e -n: Accounts with specific event types

-e -d: Accounts with specific deals

Note: If you do not specify one of these options when using the **-e** parameter, the utility returns **0**.

The **pin_rerate** utility prints the output of this option on the screen or to the **pin_rerate.pinlog** file in **\$PIN_LOG/pin_rerate**.

This parameter is optional.

Note: You cannot use this parameter with the **-m** or **-g** parameters.

-h | help

Displays the syntax and parameters for this utility.

-l [d | s | sd | n]

Specifies whether to generate a report. Options:

-ld: Specifies a detailed report (**pin_rerate.detail**).

-ls: Specifies a summary report (**pin_rerate.summary**).

-lsd: Specifies both detailed and summary reports. This is the default.

-ln: Specifies no report.

Note: When Pipeline Manager is running, you can specify these report options at rerate time (when you use the **-rerate** parameter). You cannot specify them when you create the rerate jobs.

For more information, see "[Reports Generated by the pin_rerate Utility](#)".

Parameters for Processing Rerate Jobs

You use different parameters to process rerate jobs depending on whether you rerate real-time-rated and pipeline-batch-rated events concurrently or, if you do not use Pipeline Manager for batch rating, rerate only real-time-rated events.

- [Processing Rerate Jobs When Rerating Real-Time and Pipeline Rated Events Concurrently](#)
- [Processing Rerate Jobs When You Do Not Use Pipeline Manager](#)

Processing Rerate Jobs When Rerating Real-Time and Pipeline Rated Events Concurrently

To process rerate jobs when rerating real-time and pipeline rated events concurrently, you run the following commands in the order shown.

```
pin_rerate -process jobs [-reason comma_separated_reason_codes]
pin_rerate -process queue
pin_rerate -rerate [-l [ d | s | sd | n ] ]
pin_rerate -process queue
pin_rerate -process recycle
```

Important: You must run **pin_rerate** with all of the above parameters to complete the rerating process.

These commands process rerate jobs that were created both manually, by running **pin_rerate** to select accounts for rerating (see "[Parameters for Selecting Accounts for Rerating](#)"), and automatically, by BRM automatic rerating features.

The parameters specify the following:

-process jobs [-reason comma_separated_reason_codes]

Searches for all rerate jobs with status equal to NEW, and notifies Pipeline Manager to lock the accounts in the rerate jobs. Then updates the status of the rerate jobs to WAITING_ACCOUNT_LOCK.

If you use the **-reason** parameter, performs the same actions but only for the rerate jobs associated with the specified reason codes, where:

comma_separated_reason_codes

specifies the codes you assigned as the rerate reason for the jobs you want to process. For more information, see "[Assigning Rerate Reasons to Rerate Jobs](#)".

Note: If you do not specify a rerate reason code, all rerate jobs in the rerate queue are processed.

For more information, see "[Processing Rerate Jobs According to a Rerate Reason Code for Concurrent Rerating](#)".

-process queue

Processes acknowledgment events from Pipeline Manager and updates rerate job status as follows:

- If acknowledging notification to suspend EDRs, it updates the status to ACCOUNT_LOCKED.
- If acknowledging notification to resume processing EDRs, it updates the status to READY_TO_RECYCLE.

This parameter is used twice: before and after the **-rerate** parameter.

-rerate [-l [d | s | sd | n]]

Rerates events for the accounts referenced in the rerate jobs, and updates the rerate job status to RERATED.

-l

Specifies to generate a rerating report. Options are:

-ld: Generates a detailed report (**pin_rerate.detail**).

-ls: Generates a summary report (**pin_rerate.summary**).

-lsd: Generates both detailed and summary reports. This is the default.

-ln: Generates no report.

For more information, see "[Reports Generated by the pin_rerate Utility](#)".

-process recycle [-db database_id]

Recycles the EDRs suspended during the rerating process, and updates the rerate job status to COMPLETE.

-db database_id: Specifies the ID of the database schema that contains the suspended EDRs in the format *x.x.x.x* (for example, 0.0.0.2). Use this only if the suspended EDRs are not stored in the current BRM schema.

Note: If you do not specify the **-db database_id** parameter, the suspended EDRs are picked up from the current schema.

For more information about processing rerate jobs for concurrent rerating, see ["Processing Rerate Jobs for Concurrent Rerating"](#).

Processing Rerate Jobs When You Do Not Use Pipeline Manager

If you do not use Pipeline Manager for batch rating, rerate jobs for real-time-rated events are processed as follows:

- When you run **pin_rerate** with parameters that specify the accounts and events to rerate, **pin_rerate** creates the rerate jobs for those accounts and then immediately rerates their events.
- Rerate jobs that already exist (such as those created by automatic rerating) are processed with the following parameter:

-rerate [-reason comma_separated_reason_codes]

Rerates events for the accounts referenced in the rerate jobs, and updates the rerate job status to COMPLETE.

If you use the **-reason** parameter, performs the same actions but only for the rerate jobs associated with the specified reason codes.

comma_separated_reason_codes

Specifies the codes you assigned as the rerate reason for the jobs you want to process. For more information, see ["Assigning Rerate Reasons to Rerate Jobs"](#).

Note: If you do not specify a rerate reason code, all rerate jobs in the rerate queue are processed.

For more information, see ["Processing Rerate Jobs per a Rerate Reason When Rerating Only Real-Time-Rated Events"](#).

Parameter for Purging Rerate Jobs

Use the following parameter to purge rerate jobs.

-purge [-t [ss/mm/hh/] MM/DD/YYYY]

Purges rerate jobs that have a rerate job status of COMPLETE or UNSUCCESSFUL.

[-t [ss/mm/hh/] MM/DD/YYYY]

Specifies to purge rerate jobs that have had their rerate job status set to COMPLETE or UNSUCCESSFUL before the time specified.

If no time is specified, all rerate jobs that have a rerate job status of COMPLETE or UNSUCCESSFUL are purged up to the current date.

Results

If the **pin_rerate** utility does not notify you that it was successful, look in the utility log file (**pin_rerate.pinlog**) to find any errors. The log file is either in the directory from which the utility was started, or in a directory specified in the configuration file.

If rerating fails, the utility creates a report that includes the account numbers and start times for failed rerates. The report file name is **pin_rerate.status_report**, and is in the directory from where you ran the utility.