Oracle® Communications Billing and Revenue Management

Concepts

Release 12.0
E51005-06
January 2022

ORACLE

Oracle Communications Billing and Revenue Management Concepts, Release 12.0

E51005-06

Contents

4 Overview of Billing

5 Overview of Balance Management

6 Overview of Payments

7 Overview of Customer Management

Preface

This guide provides an overview of Oracle Communications Billing and Revenue Management (BRM) system components.

Audience

This guide is intended for anyone who installs, configures, administers, customizes, or uses BRM, Oracle Communications Billing and Revenue Management Elastic Charging Engine (ECE), and Oracle Communications Pricing Design Center (PDC).

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.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

1

About Billing and Revenue Management

Learn about Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

About BRM

BRM is a charging, billing, and revenue management system for communications service providers.

BRM provides these primary functions:

- *Charging* determines how much to charge a customer for service usage and recurring fees.
- *Billing* compiles balance impacts into a bill, usually every month.
- *Payment processing* requests a payment from the customer.
- *Customer management* creates subscriber accounts and manages their status.

With BRM, you can:

- **Charge customers for service usage and subscriptions.** You can configure *online charging* (for example, to rate prepaid calls in real time) and *offline charging*, which rates usage recorded in CDR files.

  Usage is rated by the Elastic Charging Engine (ECE) or Pipeline Manager. Subscription charges (for example, recurring monthly fees) are rated by the BRM server. You define charges by creating product offerings in Pricing Design Center (PDC) or price lists in Pricing Center.

  The result of charging is a *balance impact*, that is, the amount of the charge applied to the customer's balance.

  If you are using Pipeline Manager for rating and discounting events in batch and real-time, see "About Pipeline Rating" in *BRM Configuring Pipeline Rating and Discounting*.
- **Manage customer balances.** As usage is rated, customer balances are updated in real time. You can perform a variety of accounts receivable tasks on customer balances, such as making adjustments and providing refunds.
- **Create bills and collect payments.** BRM compiles all balance impacts for customers into bills. You can use a variety of payment methods, such as credit card or email invoice, to request payments.
- **Manage customer accounts.** Accounts are stored in the BRM database. You can create, inactivate, and close accounts and manage the customer's purchased product offerings.

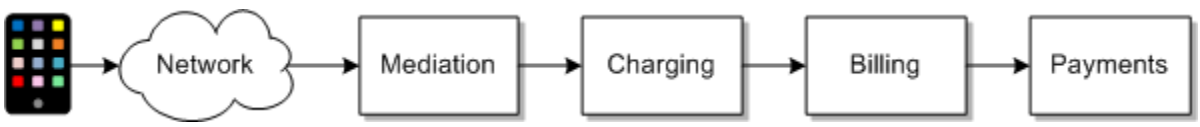
- **Collect business intelligence.** You can use BRM reports to analyze customer usage and plan your product offerings.

The following is a typical BRM subscriber scenario:

1. A subscriber purchases a package. A *package* includes one or more charge offers. A *charge offer* contains the charges that determine the cost of a service. For example, a charge offer for a mobile phone service might include the following charges:
   - A setup charge
   - A monthly recurring charge for a subscription
   - Usage charges for phone calls
2. When the subscriber purchases a package, an account is created in the BRM database. The account is associated with a variety of data, such as the following:
   - The subscriber's balance.
   - The charge offers that the subscriber purchased.
   - The services that the subscriber can use, including login names and passwords.
   - The subscriber's contact information.
3. The subscriber uses the service that she purchased, and BRM rates the service usage. For example, when a subscriber uses a post-paid mobile phone service, BRM rates the call as follows:
   a. Data about the call is received in a CDR file by Oracle Communications Offline Mediation Controller. Offline Mediation Controller performs mediation and normalization to prepare the call data for rating.
   b. Offline Mediation Controller sends the call data to ECE.
   c. Using the pricing data from the subscriber's charge offers, ECE rates the call and sends information about the call (such as the call record and its balance impact) to the BRM database.
   d. BRM updates the subscriber's balance.
4. When the subscriber is billed, typically every month, BRM compiles all the balance impacts into a bill.
5. From the bill, BRM prepares a payment request, which is handled either automatically by BRM (such as by credit card) or manually by a check-processing system.
6. When the payment is processed, the subscriber's balance is updated to show that the payment has been made.
7. A record of all of the activities described above is stored by BRM and can be compiled into reports.

Figure 1-1 is an overview of the primary BRM business functions:

Figure 1-1 Overview of BRM Business Functions



In this figure, a customer has made a call. The call is handled by a mediation system, which connects to ECE to perform rating. After rating the call and applying the charge, ECE updates the customer's balance in the BRM database. When you run billing, BRM creates a bill that includes the charges collected in the customer's balance. BRM also creates a payment request and sends it to the customer.

About Implementing BRM

When you implement BRM, you perform the following primary configuration tasks:

- **Create product offerings.** This involves creating product offerings, such as charge offers that define how customers are charged for usage, recurring charges, and one-time purchases.
- **Configure usage charging.** This involves configuring ECE to collect CDRs for offline charging, and to connect to prepaid networks for online charging.
- **Configure billing.** This involves configuring how often to run billing, as well a number of configuration options.
- **Configure revenue management.** This involves configuring payments and defining how to manage accounts receivables.
- **Configure customer management.** This involves configuring how to create customer accounts, and other configuration tasks.

About Implementing Charging

To implement charging, you use PDC to create product offerings that specify how to charge for services (for example, charge 10 cents per minute). After the product offerings are created, you configure ECE to rate usage and apply the balance impact to the customer's balance.

> **Note:**
>
> Alternatively, you can create price plans in Pricing Center and set up charging through Pipeline Manager. See Pricing Center Help and *BRM Configuring Pipeline Rating and Discounting.*

You start by defining how to sell and charge for your services in PDC:

1. Define the services that you sell, such as GSM, SMS, or broadband.

   For example, the definition of the GSM service enables you to store the customer's phone number, bearer service, and information about supplementary services, such as call forwarding. BRM includes service definitions for common communications services.

2. Define the events you want to rate. An *event* is an online action, such as a phone call, an SMS message, or a data download. Data about the event, such as when it occurred, is recorded in the BRM database. Events are also recorded for customer actions, such as purchasing a charge offer or inactivating a service. BRM supports many standard events that you can customize.

When you configure charging, you specify the chargeable events for each service. You can then apply a different charge for each type of event. For example, you can apply a charge for a data download event and a charge for a call event.

3. Create the product offerings that you sell to your customers. A customer purchases a service such as a mobile telephony service by purchasing a package, which includes one or more charge offers. Each charge offer defines the charges for usage events (such as phone calls), subscription events (such as monthly fees), and purchase events.

   ECE uses the charges defined in each customer's charge offers to determine how much to charge. If a customer changes her charge offer, her usage is rated according to the charges defined in the new charge offer.

After your pricing components are defined, you can configure usage charging in ECE. The most important part of implementing ECE is configuring the interface between ECE and the network. To do so, you use a mediation controller:

- Use Diameter Gateway, an ECE component, to manage online charging. *Online charging* rates events in real time, such as during a prepaid call. ECE can track the customer's balance and alert the customer when top-ups are needed.
- Use Offline Mediation Controller to manage offline charging. *Offline charging* is used for batch rating of events, typically from post-paid telephone usage. Offline Mediation Controller performs a number of mediation and normalization tasks, such as checking for duplicate calls.

BRM adds usage charges to customer balances throughout the billing cycle. In addition to applying usage charges, BRM applies recurring charges. For example, to charge a monthly fee for a service, you define a monthly recurring charge.

Recurring charges are not applied by ECE; instead, they are applied by BRM billing utilities. Applying a recurring charge at the same time that the customer is billed ensures that the charge is included in the latest bill.

About Implementing Billing and Payments

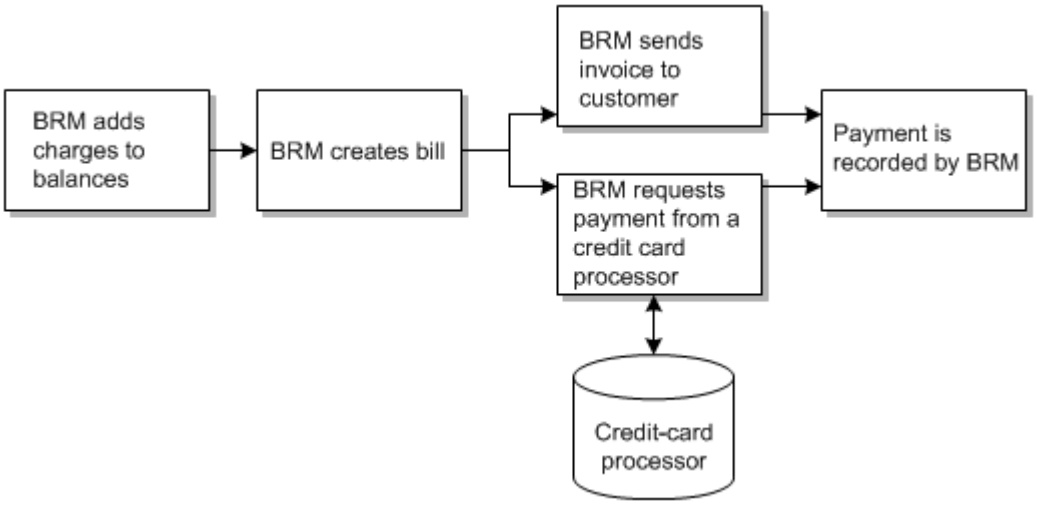
To create bills, you run a set of billing scripts, which in turn run billing utilities. For example, the **pin_bill_day** script runs several billing utilities, including the **pin_bill_accts** utility, which finds accounts that need to be billed and creates a bill for each of those accounts.

After the bill is created, BRM performs *payment processing*. BRM processes two basic types of payments:

- *BRM-initiated*, such as payment by credit card. BRM handles the interaction with the credit-card service and records the payment in the BRM database.
- *Externally initiated*, such as payment by check. BRM sends an invoice to the customer, who then renders payment manually. An accountant uses Billing Care or Payment Tool to record the payment in BRM.

Figure 1-2 is an overview of the BRM billing and payment process.

Figure 1-2 Overview of the BRM Billing and Payment Process



Implementing billing and revenue management primarily consists of configuring how you bill your customers, collect payments, and manage accounts receivable.

When you configure billing, you specify business policies such as the following:

- **What day of month to bill customers.** Each customer account has a *billing day of month (DOM)*, typically the day on which the account was created. For example, if an account is created on January 15, its billing DOM is the 15th day of each month. You can, however, specify any day for an account's billing DOM. For example, you might prefer to bill all customers on the same day or all customers of a certain category on the same day.
- **How often to bill customers.** By default, customers are billed every month, but you can change it to bill customers at any interval, such as every three months.
- **Billing business policies.** For example, specifying the minimum amount required to create a bill.

In addition to configuring billing and payments, you configure the following revenue management functions:

- **General ledger reporting.** You can assign general ledger codes to each type of balance impact and run monthly reports showing billed and unbilled revenue.
- **Invoicing.** You can design the invoice appearance and customize the content.
- **Payments.** You can set up credit-card processing and configure business policies for payments, such as how to handle missed payments.
- **Revenue assurance.** You can analyze revenue assurance data to find revenue leakage in your system.
- **Collections.** You can specify the criteria for implementing a collection process for overdue payments.

About Managing Customers

BRM stores a record of every customer, called an *account*, in the BRM database. An account includes the following:

- The customer's name and contact information.
- The account and service status (for example, Active or Inactive).
- The charge offers that the customer owns.
- The customer's balance.
- One or more *bill units* that store the charges for billing.
- The customer's payment method (for example, invoice or credit card).

To create and manage accounts, you use either the Billing Care application or the Customer Center application. Common customer management tasks include the following:

- Creating customers
- Upgrading services for customers
- Managing customer disputes
- Making adjustments to a customer's balance
- Changing account status.

You can manage groups of customers in the following ways:

- Create hierarchical groups of parent and child accounts to display relationships between accounts and to provide a mechanism that custom reports can use to identify related accounts and analyze those relationships. To enable group members to pay for other members, set up a bill unit hierarchy with the accounts' bill units.
- Create sharing groups. Members of the groups can share discounts, such as included minutes, and charges.

2

BRM System Overview

Learn about the Oracle Communications Billing and Revenue Management (BRM) system architecture.

Topics in this document:

About the BRM System Architecture

The BRM system architecture consists of:

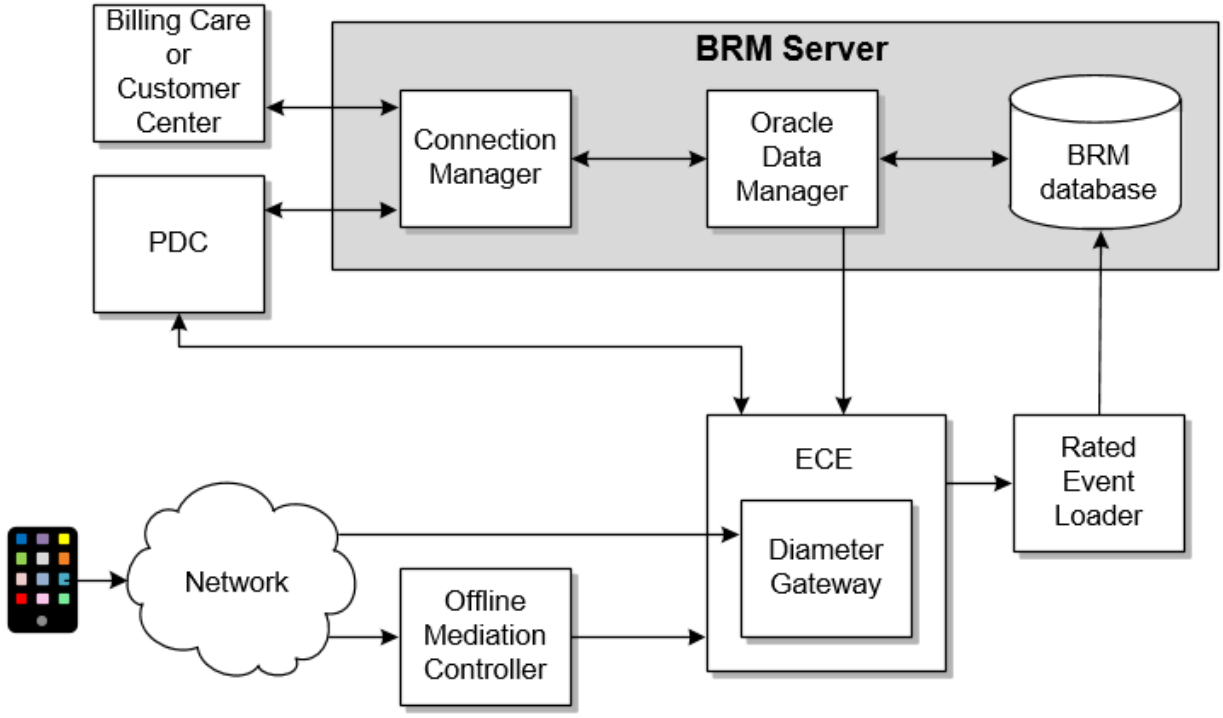
- Client applications, such as Billing Care, Customer Center, Pricing Center, and PDC.

  > **Note:**
  >
  > Pricing Center can only be used with Pipeline Manager. It cannot be used in a system with ECE.

- The BRM Server, which includes these components:
  - Connection Managers (CMs) which receive connections from client applications, and run the BRM business logic.
  - Data Managers (DMs) which handle connections to the BRM database, as well as optional databases such as a tax database.
  - The BRM database, which stores all of the BRM data.
- ECE, the BRM charging engine.
- Pipeline Manager, which rates and discounts events in batch and real-time. If you are using Pipeline Manager, see "About Pipeline Rating" in *BRM Configuring Pipeline Rating and Discounting*.

Figure 2-1 shows the components in a typical BRM system.

Figure 2-1 BRM System Components



In this figure:

- Billing Care and Customer Center are BRM clients. You use one of them to connect to BRM through the Connection Manager (CM). In a typical BRM installation, the CM handles many client connections. For example, running a billing utility from a command line connects to a CM to access the BRM system.

  In addition to providing client access to BRM, CMs run the business-logic software that performs BRM functionality. For example, the CM runs the software that creates accounts, changes passwords, and runs billing.

- The CM connects to the Oracle Data Manager (DM). The Oracle DM typically handles connections from multiple CMs. In addition to connecting the CM to the BRM database, it also translates the BRM system language into SQL commands that the database can understand.

  A typical BRM system includes additional Data Managers, each of which connects the BRM database to other clients and databases. For example, the Vertex Data manager connects BRM to Vertex, to perform tax calculations.

- PDC sends pricing data to the BRM database. PDC also sends pricing data to ECE. ECE uses the pricing data to rate usage.

> **Note:**
>
> PDC can also send pricing data to Pipeline Manager if your system is configured to do so. If you are using Pipeline Manager, see "About Pipeline Rating" in *BRM Configuring Pipeline Rating and Discounting*.

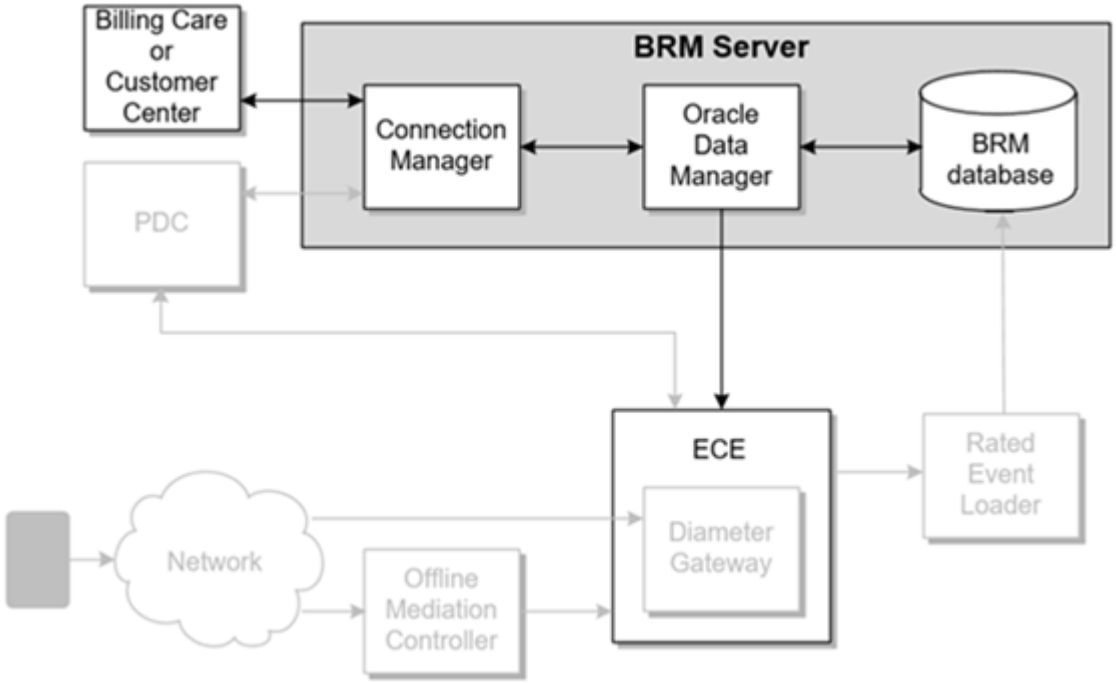
- When a subscriber makes a phone call, the call is processed by a mediation system:
  - Prepaid calls are handled by the ECE Diameter Gateway. Diameter Gateway runs as part of the ECE system.
  - The CDRs recorded from postpaid calls are handled by Offline Mediation Controller. Offline Mediation Controller is a client application of ECE.
- ECE rates the events and applies the charges to the customer balances that are stored in ECE. ECE also sends data about the rated events to BRM by using Rated Event Loader.
- Rated Event Loader sends the data about rated events to the BRM database, and update the customer's balance in the BRM database. This ensures that the balance is synchronized in ECE and the BRM database.

The following sections describe how these components are used by different business processes.

Creating an Account

Figure 2-2 shows the BRM components that are used for creating an account.

Figure 2-2 BRM Components Used for Creating an Account



To create an account:

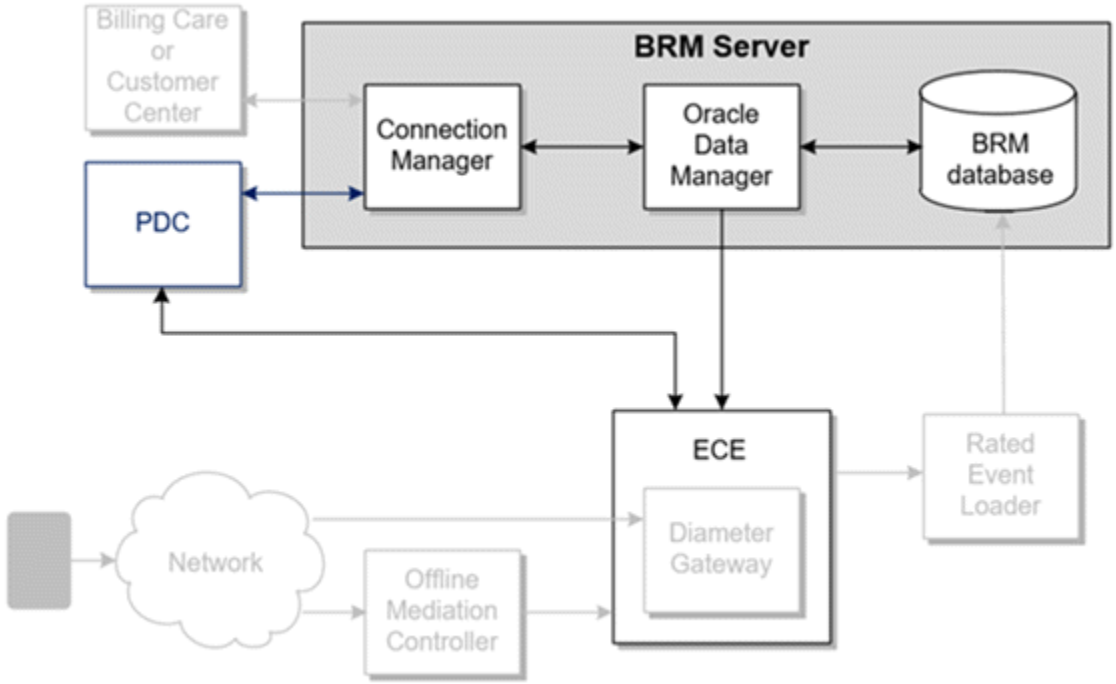
1. A CSR creates an account in Billing Care or Customer Center.
2. Billing Care or Customer Center connects to the CM.
3. The CM runs the business-logic software that creates the account, configures bill units, associates the payment option, and so on.
4. The CM sends the customer data to the Oracle DM, which translates the data into SQL commands that can be read by the database.

5. When the account is created in the BRM database, two things happen:
   - A message is sent back to Billing Care or Customer Center to inform the CSR that the operation completed successfully.
   - The EAI Java Server, initiated by the CM, sends a message to ECE and adds the new account to the ECE data cache. ECE can now rate usage for the account.

Components Used for Setting Up Product Offerings

Figure 2-3 shows the components used for setting up product offerings.

Figure 2-3 Components Used for Pricing



In this figure:

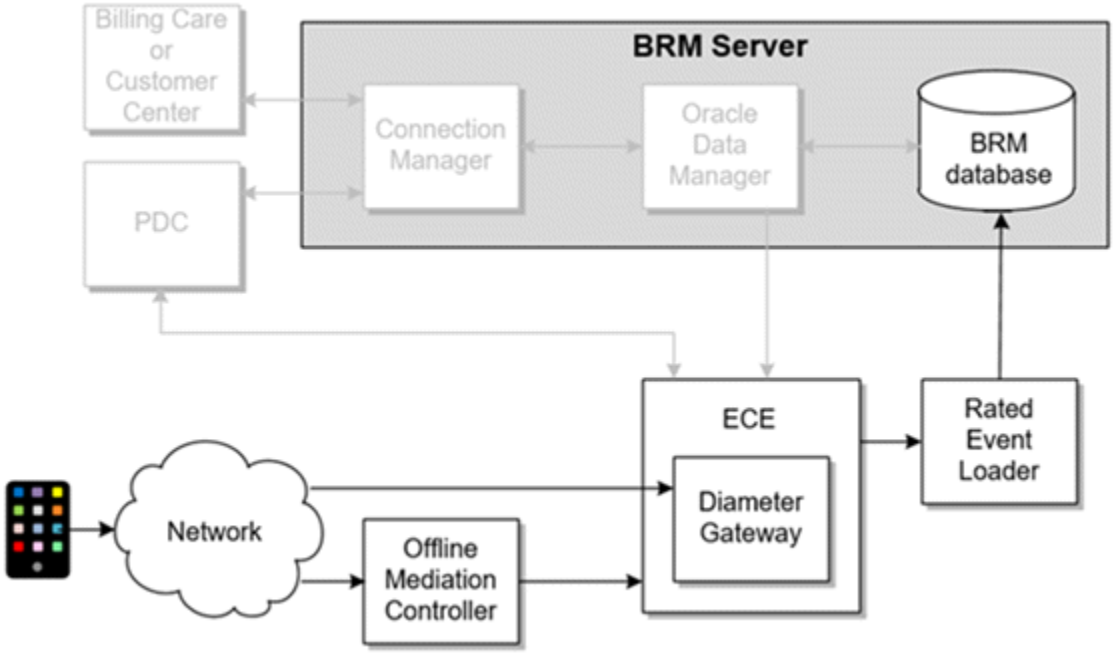
1. You create pricing components in PDC.
2. PDC sends updated pricing data to the BRM database and to ECE:
   - PDC loads pricing data into the BRM database through the CM and DM.
   - PDC uses the ECE Pricing Updater utility to load pricing data into the ECE data caches.

> **Note:**
>
> If you are using Pricing Center to create your pricing components, see Pricing Center Help.

Components Used for Usage Charging in ECE

Figure 2-4 shows the components used for usage charging.

Figure 2-4 Components Used for Usage Charging



To charge for usage:

1. The customer makes a call, which is handled by one of the mediation controllers.
   - For a prepaid call, Diameter Gateway manages the call. It carries out authorization and re-authorization by obtaining balance information from ECE customer data.
   - For a post-paid call, Offline Mediation Controller receives a record of the call and prepares it for rating. ECE then rates the call.
2. After the call is rated, ECE uses Rated Event Loader to send a record of it to the BRM database and updates the customer's balance.

If you are using Pipeline Manager for rating and discounting events in batch and real-time, see "About Pipeline Rating" in *BRM Configuring Pipeline Rating and Discounting*.

How BRM Handles Data

If you configure BRM, you need to know how BRM handles data, so you can customize BRM functionality. There are two primary elements that manage BRM data:

- *Opcodes*, which run the operations that control all of the BRM data. For example, when Billing Care or Customer Center creates an account, it uses the PCM_OP_CUST_CREATE_CUSTOMER opcode to create an account.

- *Objects*, which store the BRM data. For example, when BRM creates an account, the customer's name and contact information is stored in the BRM database in an **/account** object.

Opcodes are packaged in Facilities Modules (FMs), which are libraries run by CMs. When a BRM client connects to a CM, it can use any of the FMs that are being run by the CM.

Each FM has a function (for example, the billing FM runs billing, and the customer FM creates accounts and passwords). Each opcode in an FM carries out a specific function. For example the customer FM includes the following opcodes:

- PCM_OP_CUST_POL_VALID_NAMEINFO, which validates the data provided for a customer's name
- PCM_OP_CUST_POL_VALID_PASSWD, which validates a password
- PCM_OP_CUST_SET_NAMEINFO, which updates the customer name
- PCM_OP_CUST_SET_PASSWD, which updates a password

In these examples, two types of opcodes are shown, standard opcodes and policy opcodes. The policy opcodes include POL in their name. Policy opcodes are customizable, but standard opcodes are not. In this example, policy opcodes are used for customizing how to validate customer name information and passwords. For example, you can specify that passwords contain certain required characters.

Opcodes handle data in BRM by using a mechanism called an flist. An *flist* is a list of data that the opcode needs to carry out its function. For example, to read an object, the opcode needs to pass an flist containing the account POID. However, the Oracle database cannot read the flist format, so the CM sends the flists to the DM to translate into SQL.

Objects are containers for the BRM data. Each object holds the data for a specific BRM entity; for example, there are **/account** objects, **/bill** objects, **/service** objects, and so on.

Each object has a set of fields, and each field holds a piece of data, or an array of data. For example, the **/account** object includes these fields:

- PIN_FLD_CURRENCY holds the currency that the account uses.
- PIN_FLD_NAME holds the custom names and contact information
- PIN_FLD_STATUS holds the status of the account, such as Active or Inactive.

Objects stored in the database are created from storable classes. A *storable class* is a template that is used for each object in the database. For example, when an account is created, BRM uses the **/account** storable class to create a new **/account** object.

You can customize the data that BRM can store by customizing storable classes. For example, if you want to store custom data in a BRM account, you can customize the **/account** storable class by adding a field. All **/account** objects would then be created with your custom field.

To manage data, each object has a unique ID, known as a Portal Object ID, or POID. POIDs allow objects to link to each other. For example, the **/account** object includes the PIN_FLD_BAL_GRP_OBJ field. This field contains the POID of the **/balance_group** object that stores the customer's account balance.

3
Overview of Charging

Learn about Oracle Communications Billing and Revenue Management (BRM) charging.

Topics in this document:

For information about creating pricing components, see *PDC Creating Product Offerings*.

For information about setting up charging, see *ECE Implementing Charging*.

About Charging

In BRM, charging follows these steps:

1. Capturing an event, such as a phone call.
2. Rating the event to calculate how much to charge for it.
3. Recording the charge in the account's balance. The charge applied to the balance is called a *balance impact*.

Configuring charging requires:

- Creating your product offerings in Pricing Design Center (PDC). *Product offerings* define how you sell services to customers. For example, an International VOIP package could include a monthly charge and usage charges for international VOIP calls.

  If you are using Pricing Center to create your pricing components, see Pricing Center Help.
- Configuring charging in Elastic Charging Engine (ECE) and in the BRM server. For example, you can configure how to manage prepaid sessions.

  If you are using Pipeline Manager for rating and discounting events in batch and real-time, see "About Pipeline Rating" in *BRM Configuring Pipeline Rating and Discounting*.

BRM performs two general types of charging:

- *Usage charging* charges for service usage, such as phone calls and data downloads.
- *Subscription charging* charges recurring fees, such as a monthly subscription fee; and one-time fees, such as a purchase fee or cancel fee.

The product offerings you create can include both types of charges. It is common for a charge offer to include usage charges and recurring charges. The differences are:

- Usage charging is performed by ECE. Usage charging can be rated by measuring the duration of an event or by recording the occurrence of an event, (and by other properties of the event, such as the location of the call origin and destination).
- Subscription charging is performed by the BRM server. Subscription charging includes recurring charges and one-time charges.

How Charging Works

To charge for an event, BRM performs the following actions:

1. Measures the event based on the information in the applicable charge offer. For example, it might measure the duration of a phone call in minutes, or a data download in megabytes.
2. Applies a charge to the resulting measurement based on specifications in the product offering.
3. Adds the charge to the customer's account balance.

To charge for events, BRM requires the following types of data:

- Data about the event being charged
- Data about the customer's balance
- Data about product offerings that the customer owns

How Charging Uses Event Data

An *event* is an online action, such as a phone call, an SMS message, or a data download. Data about the event, such as when it occurred, is rated by ECE and then recorded in the BRM database. For example, when a customer makes a phone call, an event is generated that contains information about that phone call, such as the date and time the call was made, the origin and destination of the call, and the date and time the call ended. An event can also be system-generated, such as monthly subscription fees that are applied to accounts. BRM uses the data it collects about events to calculate how much to charge customers for them.

When you configure charging, you specify which events you want to charge for, for each service. You can then apply a different charge for each type of event. For example, you can apply a charge for a data download event, and a charge for a call event.

To rate an event, BRM must measure it, and then determine the charge based on event attributes. To enable BRM to measure events, you configure *ratable usage metrics* (RUMs), which specify the units to measure and how to calculate the measurement.

Common RUMs are:

- **Duration:** Length of an event in units such as seconds or minutes.
- **Occurrence:** Number of events generated during a specified period.
- **Volume:** Size of an event in units such as kilobytes or megabytes.

Typically, a single RUM is used to measure an event. But you can also use multiple RUMs to measure an event. For example, you could charge for the occurrence of a video download, as well as charging an amount based on the duration.

After measuring an event, your system must apply a charge to the measurement. To determine the charge to apply, you can use event attributes such as:

- Date and time that the event occurred, for example, different charges for peak time and off-peak time
- Origin and destination of a call, for example:
  - If the call destination is France, use the Calls to Europe charge.
  - If the call destination is Ghana, use the Calls to Africa charge.
- Type of movie downloaded, such as classic, action, or comedy
- Quality of service, such as standard or premium
- The measurement of the event itself; for example, you can charge 10 cents per minute for the first 10 minutes and 5 cents per minute for any time over 10 minutes.

How Charging Uses Balance Data

Customer balance data can be used for quantity-based rating, and for managing online charging. For example:

- You can manage prepaid calls based on the customer's balance. For example, you can specify a threshold at which a customer is notified that the remaining prepaid balance is almost zero.
- You can track the quantity of events consumed and charge accordingly. For example, you can charge one dollar for the first files downloaded, 50 cents for the next 95 files, and nothing after the 100th file.
- You can change how a customer's service is implemented based on balance data. For example, you can automatically increase a customer's quality of service when their balance reaches a specified amount. This is known as policy-driven charging. You define the policies that govern how a service can be implemented based on the customer's service usage and balance.
- You can configure chargeshare offers to share balance impacts among users. This allows customers to share charges and discounts.

Balances are organized by different types of *balance elements*. A balance element represents one of the following:

- A currency or noncurrency asset of economic value, such as U.S. dollars or included minutes
- A counter that tracks items such as dollars spent or minutes talked

> **Note:**
>
> In previous BRM releases, a balance element was called a *resource*.

Balance elements are specified when you configure pricing in a charge. They define which balance assets are increased or decreased when the charge is used to rate an event. For example, a charge of one dollar per minute for a phone call affects the U.S. dollars balance.

How Charging Uses Product Offering Data

When a customer purchases a service such as a mobile telephony service, they purchase a package that you create in PDC. A package bundles one or more charge offers and/or discount offers. Each offer defines the charges for usage events, such as a phone call, subscription events, and purchase events.

After a customer purchases a package, they own the charge offers that they purchased. ECE uses the charges defined in each customer's offers to determine how much to charge. If a customer changes their offers, their usage is rated according to the charges defined in the new offers.

BRM keeps track of the offers that each customer owns, in the BRM database, and synchronized in ECE. When an event is rated, ECE (for usage charging) and the BRM server (for subscription charging), use the charges defined in the customer's purchased offers to calculate the balance impact of the charges.

For example, if a charge offer includes a monthly recurring fee and usage charges for phone calls:

- The BRM server looks in the BRM database to find out how much to charge for the recurring charge.
- ECE looks in the ECE cache to find out how much to charge for usage.

Product offerings define everything about how to charge for an event. An event is simply a set of data; how to charge for that data is defined in your product offerings.

If you use Pricing Center to create your product offerings, rating is done through Pipeline Manager. For more information, see Pricing Center Help and *BRM Configuring Pipeline Rating and Discounting*.

About Usage Charging

All usage charging is performed by ECE. There are two types of usage charging:

- *Online charging* rates events in real-time, such as during a prepaid call.
- *Offline charging* is used for batch rating of events, typically from post-paid telephone usage.

For both online charging and offline charging, ECE receives events as usage requests. *Usage requests* contain the event data that ECE needs for rating. For example, to rate a phone call, ECE needs the number that made the call, the start time, and the end time.

When ECE receives a usage request, it uses the data in the usage request, typically the phone number, to identify the customer, which in turn identifies the charge offer that they own that is used to rate the event. In addition, the usage request includes the data needed for rating, such as the start and end times of the event.

Each type of service and event needs to be rated differently. For example, some events are rated by measuring duration, and some by measuring volume. When you configure events and services, you create event definitions that specify the data needed for charging the event. The event definitions are sent to ECE, and are stored in an ECE cache. For each incoming event, ECE uses the event definition to choose a usage request builder that creates the usage request.

A default set of event definition data is installed with ECE. If you create custom services and events, you can enrich event definitions in PDC and create customized event definitions.

Usage requests are created when network mediation clients submit data to the ECE Client:

- For online charging, a real-time online event, such as a prepaid call, is routed from the network to the Diameter Gateway, which uses the ECE Client to create a usage request. ECE processes the usage request, authorizes the call, and sends a usage response back to the Diameter Gateway. As the call is in progress, ECE also manages the interaction with the network for handling update requests, re-authorizations, and top-ups. After the call has ended, ECE rates the call.

  In addition to prepaid calls, online charging can be used for any service that the subscriber connects to and uses in real time, such as broadband access, digital content, streaming radio, and cable television.

- For offline charging, call detail records (CDRs) are processed by Offline Mediation Controller, which handles mediation tasks and normalization. Offline Mediation Controller acts as an ECE client application to create a usage request, which ECE uses to rate the events.

  Offline charging is used for batch rating of events, typically from post-paid telephone usage. Offline Mediation Controller performs mediation and normalization tasks, such as checking for duplicate calls and assembling calls that arrive in multiple records.

After an event is rated, ECE sends the rated event data to the BRM database, and the customer's balance is updated in both ECE and in the BRM database. The same process is used for loading online charging events and offline charging events.

About Online Charging

The following procedure describes how ECE typically processes a usage request for online charging.

1. The Diameter Gateway receives charging events from the network.
2. The Diameter Gateway creates a usage request.
3. ECE calculates the charge of the event. For prepaid calls, ECE can perform the following tasks:
   - Calculate the cost of the call and compare it to the amount available in the customer's balance.
   - Send a *usage response* to the network to either allow the call to start, or to inform the network that the customer's balance is too low to make the call.

     A usage response can contain an *in-session notification* about the active customer session. For example, ECE can send an in-session notification about a customer's credit limit threshold breach information. The network mediation system can use the data from the in-session notification for sending a message to the customer.
   - Send notifications to the network when a customer's balance needs to be topped up.
   - Update the customer balance as balance elements are consumed.
4. When the call ends, ECE updates the customer's balance and sends a record of the event to the BRM database.

During online charging, ECE communicates with the network components through the Diameter Gateway to manage update requests, re-authorization requests, balance queries, and top-ups. You can also use online charging to manage the following functions:

- **Policy-driven charging.** You can change a customer's network configuration based on their service usage. For example, you can track the amount of data that a customer downloads and change their download speed when the amount of usage reaches a threshold amount.
- **Multiple Services Credit Control (MSCC).** When multiple services are used within the same user session, ECE can calculate the charge for each service separately.
- **Server-initiated re-authorization requests.** Re-authorization requests are typically generated by the network at regular intervals, but you can configure ECE to initiate a re-authorization request when a change occurs in a customer's account or balance data that might cause an in-session even to be charged different.
- **Advice of Charge (AoC).** ECE can use AoC to tell a customer about the charge for a service at the beginning of a session, during a session or at the end of a session.
- **Advice of Promotion (AoP).** ECE can use AoP to notify a customer that a better price could be obtained for the service they are about to use if they use the service at a different time.

About Offline Charging

Offline charging is used for batch rating of events, typically from post-paid telephone usage. Offline Mediation Controller performs mediation and normalization tasks, such as checking for duplicate calls and assembling calls that arrive in multiple records.

The following procedure describes how ECE typically processes a usage request for offline charging.

1. Offline Mediation Controller collects charge detail records (CDRs). Depending on the type of service, the records can be in files, or can be received directly from a network application.
2. Offline Mediation Controller converts the event records into its own format, and performs processing such as duplicate checking and aggregation.
3. Offline Mediation Controller sends the records to ECE as usage requests.
4. ECE calculates the charges based on the request attributes. ECE supports debit and refund operations for which the charges are passed in the input.

   If ECE detects errors (for example, it does not find a valid offering for charging for the request), then it rejects the request. It sends a negative usage response to Offline Mediation Controller which then rejects the CDR.
5. ECE manages the current state of customer balances within the charging system so that it is synchronized with the balance information in BRM.

Offline Mediation Controller performs several mediation and normalization tasks:

- Checks for duplicate records and duplicate input files.
- Ensures that records and files are processed in the correct order (called sequencing).
- Suspends failed records and recycles them after they have been corrected.
- Aggregates records to manage charging more efficiently. For example, you can aggregate records by time or by volume.

- Enhances and normalizes data, for example:
  - Mask social numbers that customers do not want on their bill.
  - Normalize number prefixes.
  - Map multiple service codes into a single service code.
- Filter records based on data in the records, for example, discard records from specified APNs.

For more information about preparing records for offline charging, see *Offline Mediation Controller User's Guide*.

About Subscription Charging

Subscription charging is performed by the BRM server. These charges are always based on the occurrence of an event generated by the BRM server, or an event triggered by changes to a customer's product offerings. For example:

- When you run billing, BRM creates events that trigger recurring charges, such as a monthly service fee.
- When a customer purchases a charge offer, BRM can charge a purchase fee.
- When a customer cancels a charge offer, BRM can charge a cancelation fee.

As with usage charges, subscription charges are defined in the product offerings that a customer purchases. For example, when a customer purchases a charge offer that includes a recurring charge, the charge offer defines the following:

- The amount to charge
- How often to charge (for example, monthly or yearly)
- If the charges should be prorated if the charge offer is valid for only part of the time charged for.

Recurring charges are based on cycles; for example, a monthly recurring charge has a one-month cycle. Therefore, recurring charges are often referred to as cycle charges or cycle fees.

> **Note:**
>
> Because you can define recurring charges to use any length of cycle, you need to consider the impact of revenue recognition in general ledger (G/L) reports. For example, if a customer pays a monthly fee in advance for the following month, that fee cannot be included as earned revenue until the end of that month.

There are three types of recurring charges:

- A *cycle forward charge* applies a charge in advance, such as a monthly charge applied at the beginning of the month.
- A *cycle arrears charge* applies a charge to the previous month, such as a monthly charge applied at the end of the month.
- A *cycle forward arrears charge* applies a charge at the end of the month, but the balance impact is recorded at the beginning of the month. The customer is not charged until the

end of the month, but the balance impact is made at the beginning of the month, and can be recognized as unearned revenue at the beginning of the month.

About Product Offerings

You create product offerings to define the services you offer and how much to charge for those services. BRM uses the charges defined in the product offerings to determine how much to charge for the use of your services. You can use the following types of charges:

- **Usage:** Charges for the use of a service, such as telephone calls or broadband sessions.
- **Recurring:** Ongoing charges that are not generated or affected by usage, such as a monthly subscription fee.
- **One-time:** Nonrecurring charges, such as setup or cancellation fees.
- **Rollover:** Charges that extend the validity of unused balances to succeeding cycles. For example, included minutes are often rolled over.
- **Fold:** Charges used to zero-out a balance or convert one balance into another. For example, you could configure a fold charge to convert frequent flyer miles to a dollar amount.

Product offerings in PDC are represented in the following pricing components.

> **Note:**
>
> Pricing Center uses different terms for these pricing components. In Pricing Center, product offerings are called *price lists*, bundles are called *deals*, and packages are called *plans*. For more information, see Pricing Center Help.

- **Charge Offers and Discount Offers:** Define how to charge for a service. For example, a charge offer for a mobile phone service might include criteria for calculating the following charges:
  - A setup charge
  - A monthly subscription charge
  - Usage charges for phone calls

  A discount offer for a mobile phone service might include criteria for calculating the following discounts:
  - 50% off the monthly charge for the first 6 months
  - 25% off for usage over 750 minutes
  - Reduction in usage charges if the subscriber has included minutes
- **Bundles:** Contain one or more charge offers, discount offers, or both. You can reuse charge offers and discount offers in multiple bundles.

  For example, you might mix and match charge offers and discount offers for a mobile phone service to create the following bundles:
  - **Bundle A: Basic Voice**
    - **Charge Offer A:** $300 setup fee

    - **Charge Offer B:** $50 monthly fee
    - **Charge Offer C:** $0.50 per minute for all minutes over 300 per month
    - **Discount Offer A:** 25% off usage over 750 minutes per month
  - **Bundle B: Promotional Voice**
    - **Charge Offer A:** $300 setup fee
    - **Charge Offer D:** $20 monthly fee
    - **Charge Offer E:** $1 per minute for all minutes over 300 per month
    - **Discount Offer A:** 25% off usage over 750 minutes per month
    - **Discount Offer B:** No setup fee if the service is purchased before a specified date
    - **Discount Offer C:** 50% off monthly fee for the first 6 months

  All offers in a bundle must be associated with the *same* service.

- **Packages:** Contain one or more bundles. You use packages to offer services to customers. To subscribe to your services, a customer purchases a package.

  For example, if your company provides broadband access and VoIP (voice over Internet protocol), you might create the following packages:

  - **A package that offers only broadband access.** This package would contain one or more broadband access bundles, such as a high-speed broadband access bundle and a mobile broadband access bundle.
  - **A package that offers only VOIP.** This package would contain one or more VOIP bundles, such as a standard calling bundle and an international calling bundle.
  - **A package that offers broadband access and VOIP.** This package would contain at least one broadband access bundle and at least one VOIP bundle.

Figure 3-1 shows how product offerings are organized in BRM. To create product offerings, you start by creating charge offers and discount offers. You then organize them in bundles and packages.

Figure 3-1 Product Offerings Created in PDC

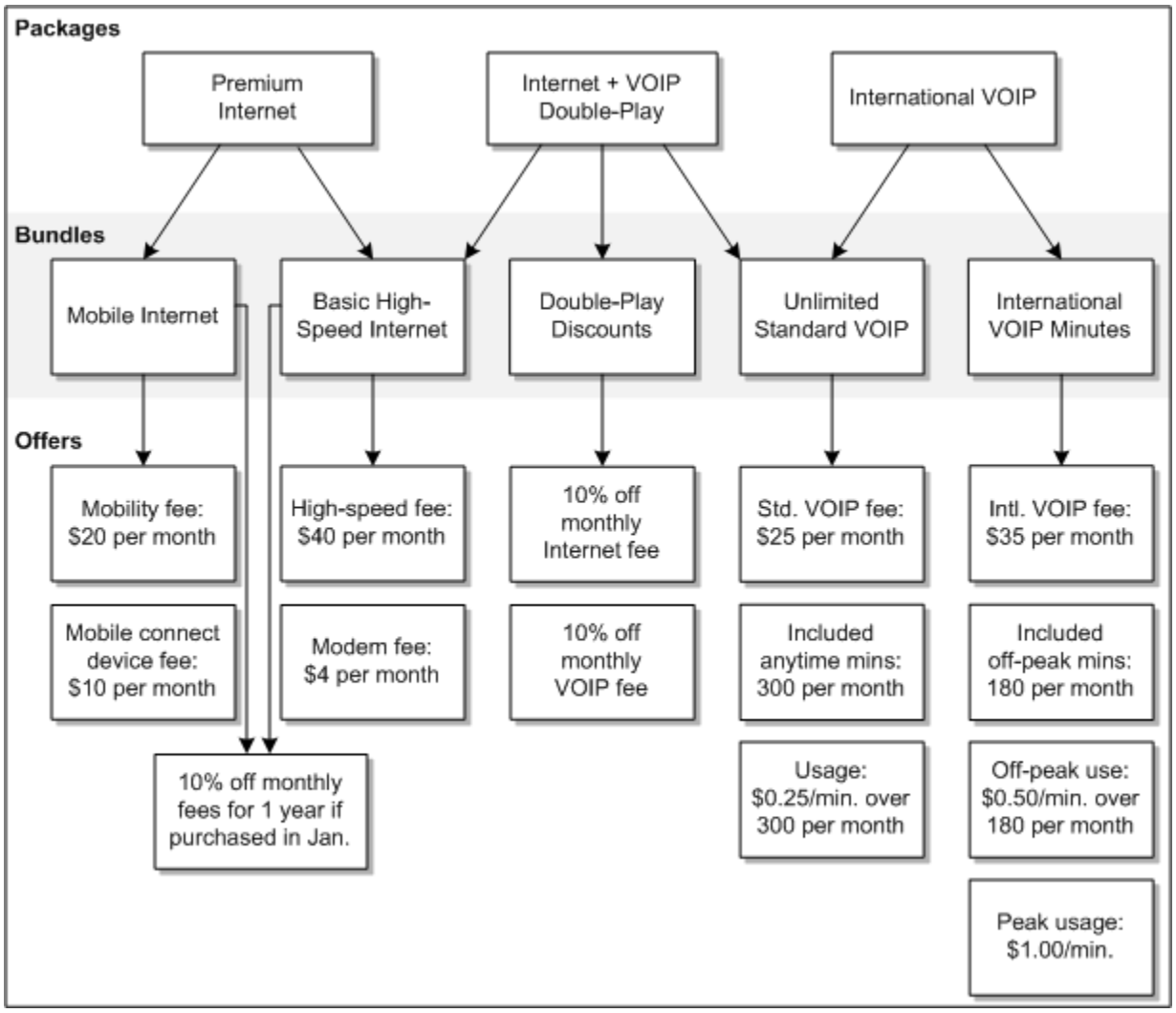
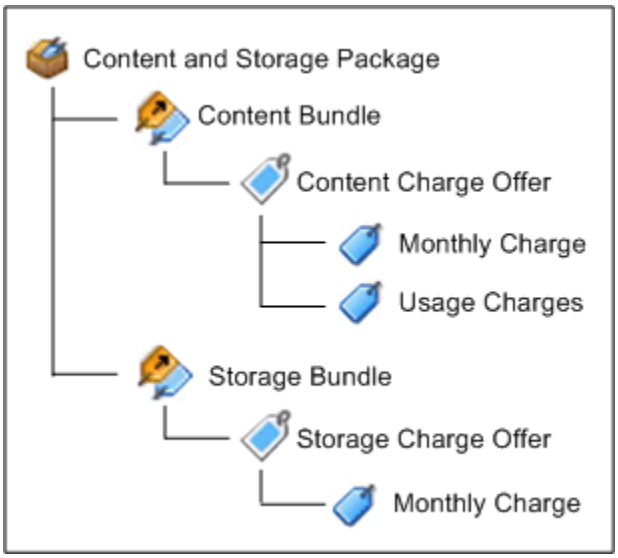


Figure 3-2 shows the hierarchy of pricing components. In this example:

- The package offers two services: content download, and cloud-based storage.
- Each service has a separate bundle.
- Each bundle has a charge offer.

Figure 3-2 Product Offering Hierarchy



About Implementing Charging

To implement charging, you need to configure and use several different BRM components. Assuming that BRM, ECE, and PDC are already installed, the overall process includes:

- Plan your product offerings. This is usually done conceptually (for example, on a spreadsheet). Planning your product offerings reveals which events and services you need to charge for.
- Create product offerings in PDC.
- Configure ECE and the BRM server to keep customer data updated in ECE. This includes:
  - Configure event notification and account synchronization in the BRM server to send data to ECE.
  - Configure the Customer Updater in ECE to receive data from BRM. This configuration is performed when you install ECE.
- Configure ECE and PDC to keep pricing data updated in ECE. This includes:
  - Configure PDC to send pricing data to ECE. To do so, you set up a JMS queue on a WebLogic server where PDC can publish pricing data for ECE and configure PDC to publish pricing data to the queue.
  - Configure the Pricing Updater in ECE to receive data from PDC. This configuration is performed when you install ECE.
- Configure ECE to alert the BRM server about changes that need to occur, such as trigger billing for an account. This includes:
  - Configure the BRM Gateway to connect to a CM as a BRM client. This configuration is performed when you install ECE.
  - In ECE, configure the JMS queues that send data to the BRM Gateway. This configuration is performed when you install ECE.
- Configure ECE for online charging. This includes:

  - Configure whether to send notifications to communicate with online charging clients, such as top-up clients and balance-query clients.
  - Configure charging rules and options, such as how to manage balance elements in a prepaid session.
- Configure ECE for offline charging. This includes:
  - Configure the ECE nodes in Offline Mediation Controller.
  - Configure Offline Mediation Controller to send usage requests to the Elastic Charging Client.
- Configure ECE to send rated events to BRM. This includes configuring the Rated Event Loader.
- Configure ECE and the BRM server to enable rerating. This includes:
  - Configuring rerating in the BRM server to specify events that need rerating, and to send events to ECE for rerating.
  - Configure the ECE EM Gateway to receive events from BRM for rerating.
  - Configure the acknowledgement queue on the BRM server to receive messages from ECE.
- Configure suspending and recycling events in Offline Mediation Controller and the BRM server.

To configure charging, see the following books:

- See *BRM Creating Product Offerings* for information about creating packages, bundles, charge offers, and discount offers. This book is for pricing analysts who need to create product offerings.
- See *ECE Implementation Guide* for information about configuring online and offline charging, rerating, and suspending and recycling events. This book is for users who implement and configure charging in a run-time environment.

4

Overview of Billing

Learn about Oracle Communications Billing and Revenue Management (BRM) billing.

Topics in this document:

For information about setting up and running billing, see *BRM Configuring and Running Billing*.

About Billing

Billing is the process of compiling charges in a customer's balance and creating a bill. The amount due in the bill is sent to the customer as a payment request.

During the time between bills, a customer's charges are stored in *bill items*. There are different types of bill items; for example, usage items that store usage charges, and cycle forward items that store recurring charges.

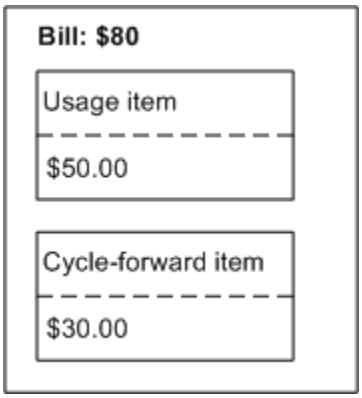
When billing is run, BRM creates a bill. A BRM bill is a **/bill** object in the BRM database that stores the charges from bill items and all billing information necessary to generate a request for payment.

> **Note:**
>
> The **/bill** object only *stores* billing information and is not a request for payment itself. You request payments from customers by creating a payment request.

All accounts in the database, including accounts that do not pay their own bills, have their own **/bill** objects. The **/bill** object stores data such as the bill creation date, total charges accumulated in the bill, the amount due from the customer, and the bill due date.

Figure 4-1 shows a bill with two bill items.

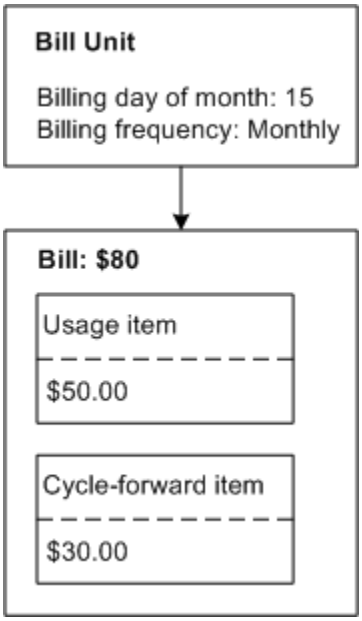
Figure 4-1 Bill With Bill Items



Each account includes at least one *bill unit* that defines the charges from each bill item that belong in a bill. BRM creates one **/bill** object for each **/billinfo** object in the BRM database.

Figure 4-2 shows a simplified bill unit and bill relationship. The bill unit defines when and how often to create a bill. The bill includes items that contain the charges collected over the month between bills. A bill is produced for every bill unit.

Figure 4-2 Bill Unit and Bill



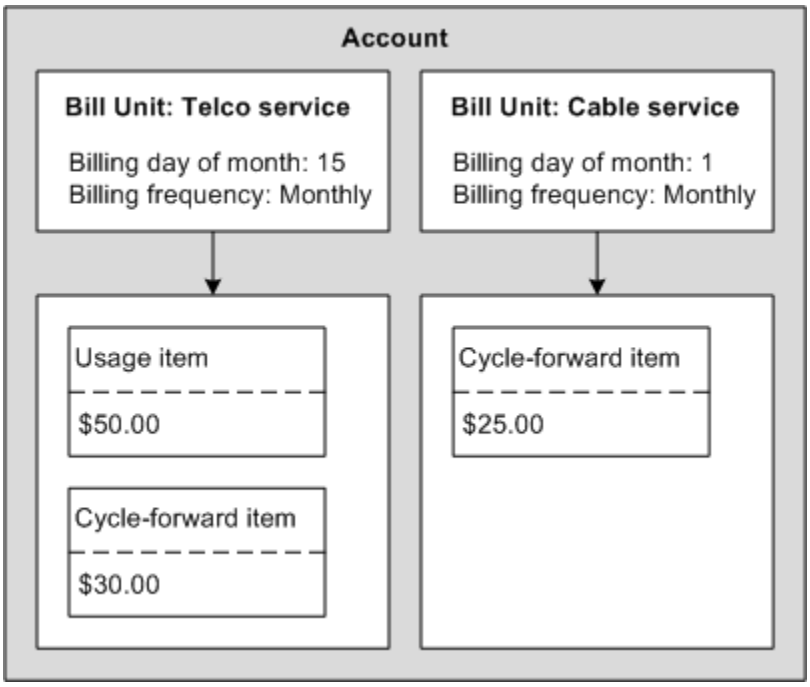
Billing is based on cycles, usually monthly. Each bill unit has a billing day of month (DOM), which is typically the day of month on which the account is created. For example, if an account is created on May 7, all of its bill units, by default, have the seventh day of the month as their billing DOM.

> **Note:**
>
> For a bill unit, the billing DOM and the accounting DOM are the same day, which is specified in the PIN_FLD_ACTG_CYCLE_DOM field of the **/billinfo** object.

In addition to the DOM, each bill unit has a billing frequency. For example, if the account is billed monthly, its bills are generated on June 7, July 7, August 7, and so on. Most customer accounts are billed monthly, but you can bill accounts at any monthly interval (for example, bimonthly, quarterly, semiannually, or annually).

An account typically has one bill unit but can have multiple bill units; for example, a bill unit for each service the customer owns. By default, all bill units in an account have the same billing DOM and billing frequency, but you can modify each bill unit to have a different billing DOM and billing frequency. Figure 4-3 shows an account with two bill units. In this example, the bill unit for the cable service has no usage fees.

Figure 4-3 Account With Two Bill Units



To create bills, you run billing by running a set of billing scripts, which in turn run billing utilities. For example, the **pin_bill_day** script runs several billing utilities, including the **pin_bill_accts** utility, which finds bill units that need to be billed and creates a bill for each of those bill units. For the account shown in Figure 4-3, billing runs on the 1st day of the month for the telco service, and on the 15th day of the month for the cable service.

After finding the bill units that need billing, BRM does the following:

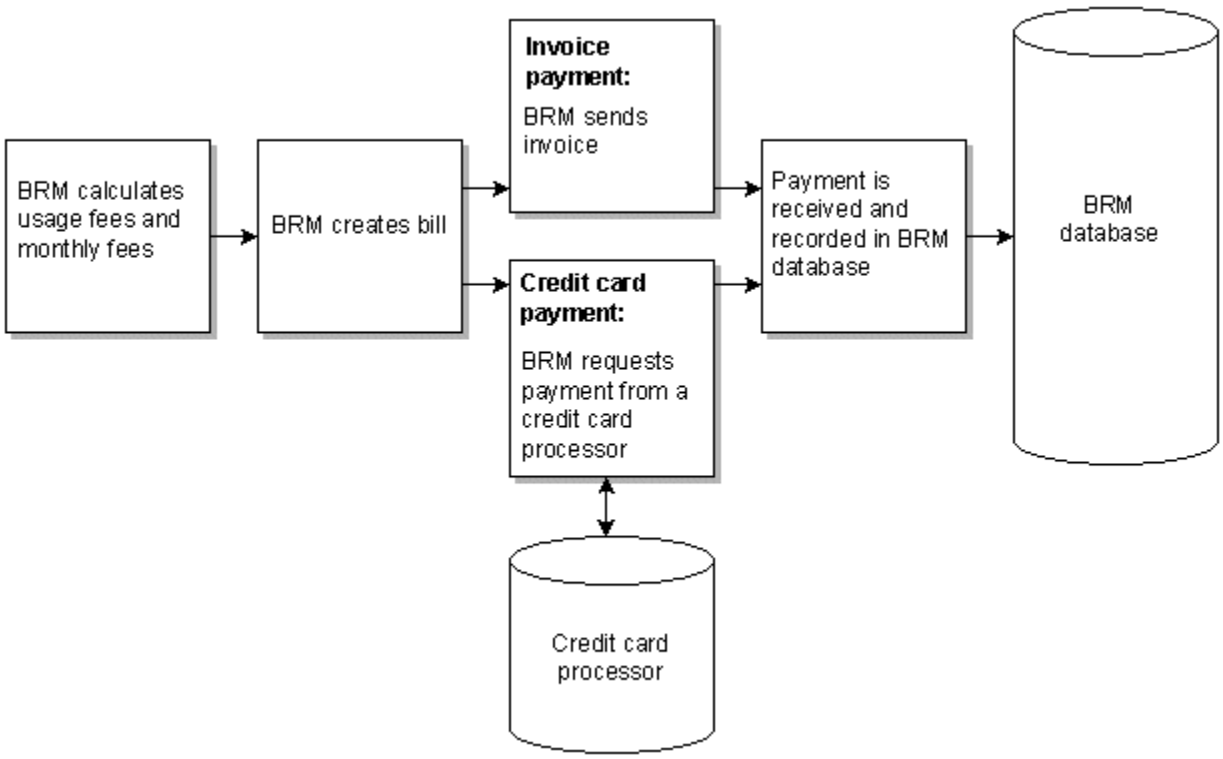
1. **Performs monthly accounting**. BRM compiles the total amount of balance impacts that have occurred in the past month. This can include usage fees and recurring fees. This monthly accounting is called the *accounting cycle*. For more information, see "About Accounting and Billing Cycles".

2. **Finalizes the bill**. To finalize a bill, BRM changes the status of all the bill items associated with the bill from pending to open so that they stop accumulating charges and so that payments can be applied to them. In addition, a payment due date is added to the bill. The time period during which charges accumulate in an account before a bill is finalized is called the *billing cycle*. (For more information, see "About Accounting and Billing Cycles".) Typically, a bill is finalized monthly, at the end of each accounting cycle. However, you can bill in any multiple of one month (for example, every two months, quarterly, or yearly). A finalized bill includes balance impacts from each accounting cycle in the billing cycle.
3. **Requests a payment**. BRM supports two types of payments:
   - You process BRM-initiated payments by automatically requesting payments from a credit card or debit card processor.
   - You process externally initiated payments by sending invoices, receiving the payments, and processing the payments in batches. An invoice lists the events that were charged for, and the customer's total balance for that bill.

   When a payment is recorded in the BRM database, the customer's account balances are updated automatically.

Figure 4-4 shows how BRM compiles bills and requests payments from customers:

Figure 4-4 Regular Billing Process in BRM



About Bill States

You use bill states to determine whether billing is run for the bill and for informational purposes in external applications.

There are two levels of bill state:

- **Bill unit-level**: Stored in the **/billinfo** object, this state determines whether billing is run for the owning account. See "About Bill States and the pin_bill_accts Utility" in *BRM Configuring and Running Billing* for more information.
- **Bill-level**: Stored in the **/bill** object, this state is informational. It can be queried and updated by external applications through the REST API.

The bill-level states are:

- **INPROGRESS**:
  - Set by BRM when the bill is created.
  - Set by an external application through the REST API to indicate that BRM should resume billing for the related bill unit of a bill that was previously in the ONHOLD state.
- **NEW**: Set by BRM when the bill is finalized.
- **PARTIALLYPAID**: Set by BRM when only part of the bill has been paid. (The value of the PIN_FLD_DUE field in the **/bill** object is greater than zero but less than the value of PIN_FLD_CURRENT_TOTAL.)
- **SETTLED**: Set by BRM when the total amount has been paid. (The value of PIN_FLD_DUE is zero and the value of PIN_FLD_CURRENT_TOTAL is a positive number.)
- **ONHOLD**: Set by external applications through the REST API to indicate that BRM should suspend billing for the related bill unit.
- **SENT**: Set by external applications through the REST API to indicate that the bill has been sent to the customer. No impact on BRM operations.
- **VALIDATED**: Set by external applications through the REST API to indicate that the bill has been validated externally. No impact on BRM operations.
- **UNDEFINED**: Set by BRM for backward compatibility on bills created before bill-level bill states were introduced.

See the Update a Customer Bill endpoint in REST Services Manager API for Billing and Revenue Management for more information about setting the bill state through the REST API.

About Accounting and Billing Cycles

Billing is performed in *cycles*. There are two types of cycles:

- The *accounting cycle* compiles all of a customer's balance impacts and stores them in bill items. The accounting cycle is always monthly.
- The *billing cycle* defines how often to request a payment for the balance impacts contained in the bill items. You can request payments every month or in any number of complete months, such as quarterly. Therefore, the accounting cycle and the billing cycle always start on the same date, but they can be different lengths.

If you bill customers every month, the accounting cycle and the billing cycle are usually the same. However, you can bill in any multiple of one month (for example, every two months, quarterly, or yearly). If a billing cycle is longer than the accounting cycle, the bill includes balance impacts from multiple accounting cycles.

Accounting cycles and billing cycles are different in other important ways:

- **Customer impact**. The accounting cycle is an internal cycle; that is, it does not affect a customer in any way other than adjusting the account balance. The billing cycle is an external cycle. After you run billing, your customers receive an invoice or receive a credit card or debit card transaction.
- **When activity occurs**. For accounting cycles, activities such as recording balance impacts into usage items occur *during* the accounting cycle. For billing cycles, activity occurs only at the *end* of the billing cycle when BRM finalizes a bill and requests a payment from the customer.

About Accounting Cycles

By default, an accounting cycle always ends at midnight, specifically at 23:59:59, and the next accounting cycle always begins at 00:00:00. BRM performs various tasks at the end of one accounting cycle and the beginning of the next accounting cycle:

At the end of an accounting cycle, BRM performs these tasks:

- Applies balance impacts from recurring charges. (See "About Subscription Charging" for more information.) This includes balance impacts from the following:
  - Cycle forward charges. BRM creates one or more cycle forward items, one for each service that the customer owns. The cycle forward items include any cycle forward balance impacts that apply to the following month. The cycle forward items have a status of pending.
  - Deferred cycle forward charges. A deferred cycle forward charge is a recurring charge that has been scheduled to take effect in a future billing cycle.
  - Cycle arrears charges. BRM applies balance impacts from cycle arrears charges to the current usage item.
  - Cycle forward arrears charges. BRM applies balance impacts from cycle forward arrears charges to the next cycle's cycle forward arrears item.
  - Cycle discount grants.
  - Monthly rollovers. Rollovers are balances, such as minutes, that can have their validity extended.
- Applies balance impacts for fold events. For example, if a charge offer uses fold events to remove unused free hours, the fold events are rated, and the balance impacts are applied at the end of the accounting cycle.
- Calculates deferred taxes, if any, and applies them as balance impacts. See "About Calculating Taxes" in *BRM Calculating Taxes*.

At the beginning of a new accounting cycle, BRM performs these tasks:

- Creates a usage item. Balance impacts from usage fees, cancel fees, and purchase fees are added as they occur. The usage item has a status of pending.
- Creates custom items created to track charges for specific services. These services are specified in the **/config/item_tags** and **/config/item_types** objects. These items have a status of pending.

If the account uses a multimonth billing cycle, new cycle forward and usage items are created every month, resulting in multiple cycle forward and usage items.

About Accounting Cycle Dates

The day that the accounting cycle starts is called the accounting day of month (DOM). By default, an accounting cycle begins on the day of the month that the customer account is created. For example, if the customer creates an account on the 15th, the accounting cycle starts on the 15th day of the month.

Although an accounting cycle is always one month long, the length of the accounting cycle changes from month to month. For example, if an accounting cycle starts on the 15th day of the month, there are more days between January 15 and February 15 than there are between February 15 and March 15.

> **Note:**
>
> For a bill unit, the billing DOM and the accounting DOM are the same day, which is specified in the PIN_FLD_ACTG_CYCLE_DOM field of the **/billinfo** object.

You can change the default accounting cycle date for all accounts to a single date, but that can result in an excessive load on the BRM system.

About Billing Cycles

Billing cycles consist of a billing DOM and a billing frequency:

- The billing DOM specifies the date on which BRM finalizes a bill and requests payment from the customer. The billing DOM is determined by the bill unit's billing segment, the DOM of the account's other bill units, the default setting in the Connection Manager (CM) configuration file, or the current date.
- The billing frequency specifies how often to finalize a bill and request payments from customers. The billing cycle length is any whole multiple of the accounting cycle. For example, a monthly billing cycle corresponds to one accounting cycle, and a quarterly billing cycle corresponds to three accounting cycles. Figure 4-5 shows the billing and accounting cycles for an account that is billed every three months.

Figure 4-5 One Month Accounting and Three Month Billing Cycles

The billing DOM and billing frequency are set in the bill unit. Accounts with multiple bill units can have different billing cycle settings for each bill. For example, an account with two bill units can have the following cycles:

- One bill unit with a billing DOM of 5 and a monthly billing frequency.
- One bill unit with a billing DOM of 15 and a quarterly billing frequency.

> **Note:**
>
> Child bill units must have the same billing DOM and billing frequency as their parent bill unit.

About Actions Performed at the End of a Billing Cycle

Typically, at the end of a billing cycle, BRM performs the following actions:

- Changes the status of the bill items associated with each bill unit to open. Therefore, no further balance impacts are added to those items, and BRM can request payments for those items.
- Finalizes a bill for each bill unit in the BRM database. If the billing cycle includes more than one accounting cycle, the bill includes balance impacts from multiple bill items. For example, a quarterly bill includes balance impacts from three usage items and at least three cycle forward items.
- Depending on the payment method, BRM either requests a BRM-initiated payment (credit card or direct debit) or creates an invoice for the bill. For some payment methods, such as Undefined, BRM makes no payment request.

Collecting payments does not occur automatically at the end of a billing cycle. You must set up billing applications that automatically request payments at the end of an account's billing cycle.

About Auto-Triggered Billing

When auto-triggered billing is enabled, BRM automatically triggers billing when an event occurs *after* the billing date but *before* billing is run. For example:

1. An account's billing date is May 15, and billing will be run for the account on May 15 at 02:00:00.
2. On May 15 at 01:00:00, one hour after the end of the previous billing cycle but one hour before that billing is run, a usage event associated with the account occurs. This usage event belongs to the next billing cycle.
3. To ensure that the usage event is recorded in the correct billing period, BRM immediately performs the billing processes (for example, changes item status to open) and finalizes a bill for the account.
4. The event that triggered billing is included in the items for the next bill.

> **Note:**
>
> Auto-triggered billing performs only the operations that the **pin_bill_accts** utility normally performs. It does not do any payment requests; those are done when you run the **pin_collect** utility.

Events that trigger billing include the following:

- Purchasing a charge offer
- Changing account status

- Canceling a charge offer
- Rating a usage event

> **Note:**
>
> An event does not need to have a balance impact to trigger billing.

By default, auto-triggered billing is *always* enabled. You can disable auto-triggered billing when, for example, you might want to run billing only by running the billing application (**pin_bill_accts**).

> **Note:**
>
> If you use delayed billing, auto-triggered billing is always enabled for the delay period. And by default, it is also enabled for the last two days *only* of each bill unit's accounting cycle.

About Delayed Billing

You can set up BRM to delay billing for accounts after the end of a billing cycle. This is called *delayed billing*. Delayed billing essentially extends a billing cycle by the delay interval. You use delayed billing to bill for events that occur within a billing cycle but are not recorded during that cycle.

For example, if a batch of events does not arrive until after the end of the billing cycle, you delay billing until all events in the batch are recorded in the BRM database. When you use delayed billing, billing for all the accounts in your BRM system is delayed for the same amount of time; you cannot specify multiple billing delay periods.

When your system is set up to use delayed billing, BRM performs partial billing to enable the new event to be rated and applied to the correct billing period. Partial billing ensures that new events impact bill items of the next billing cycle and old events impact bill items of the previous billing cycle.

For more information about delayed billing, see *BRM Configuring and Running Billing*.

About Accounting Types

BRM bills include the charges incurred during the current billing cycle and, optionally, any unpaid charges from previous billing cycles. You control whether BRM bills include charges from previous billing cycles by setting the accounting type:

- With *open item accounting*, a customer is billed only for charges from the bill items in the current bill. If a customer does not pay a bill, the next bill does not include charges for the bill that the customer did not pay.

  You typically use open item accounting for non–credit card accounts, where a customer receives an invoice. Each invoice includes the items that apply to a single billing cycle. If a customer does not pay a bill, the customer still has the invoice for the old bill when the customer receives the next invoice.

- With *balance forward accounting*, a customer's bill includes all the charges that a customer owes, including those from previous billing cycles. If a customer does not pay a bill, the next bill includes the charges from the previous bill.

  Accounts for customers who pay by credit card should always use balance forward accounting. Balance forward accounting is the default for new accounts.

Accounting types are set in the bill unit rather than in the account. This enables accounts with multiple bill units to have different accounting type settings for each bill. For example, an account with two bill units can have one bill unit with an open item accounting type and another bill unit with a balance forward accounting type.

About Multiple Bills per Cycle

An account's billing information is stored in a bill unit (**/billinfo** object in the BRM database). The bill unit associates each account balance group with a bill and a payment method. When you bill accounts, a bill is produced for every bill unit.

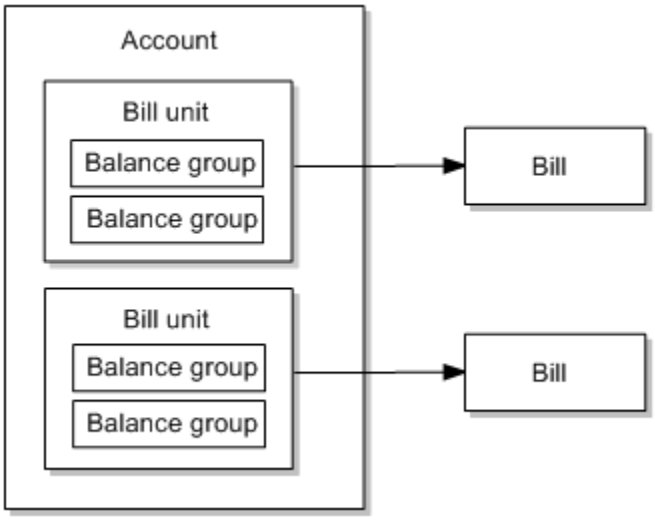
You can create multiple bill units for a single account. Each bill unit in an account has its own payment method, accounting type, billing DOM, and billing frequency.

By default, accounts are created with one bill unit. You create additional bill units by using Billing Care or Customer Center.

You can also create bill units by implementing BRM opcodes in your custom code. Specify the account to which the bill unit belongs and link the account balance groups to the new **/billinfo** object.

When an account has multiple bill units, a bill is produced for each bill unit in the account, as shown in Figure 4-6.

Figure 4-6 Multiple Bill Unit Account



You perform the following for each bill unit in an account:

- Associate a payment method, such as credit card, direct debit, or invoice. With multiple bill units, accounts can be billed for services separately, using a different payment method for each bill.

- Specify the billing DOM.
- Specify the billing frequency.
- Specify the accounting type: open item accounting or balance forward accounting.

5

Overview of Balance Management

Learn about Oracle Communications Billing and Revenue Management (BRM) balance management.

Topics in this document:

About Balances

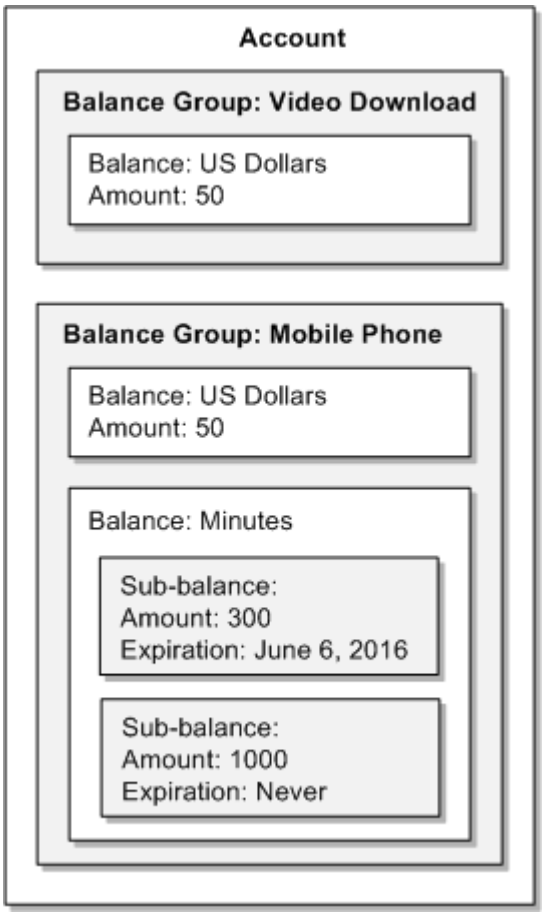
The customer balance is the amount that the customer owes or is credited with. Every account has at least one balance group that keeps a record of the customer's assets. Each balance group can include multiple balances based on different types of balance elements. For example, a balance group might contain a balance for dollars and a balance for minutes.

Each balance can include sub-balances. For example, the balance that tracks minutes might include 300 minutes that are valid only for the current month and 1000 minutes that never expire.

An account can include multiple balance groups. For example, an account might include separate balance groups for different types of services. This enables the balances to use different credit limits.

Figure 5-1 shows an account with two balance groups, one for a video download service, and one for a mobile phone service. The video download service tracks only dollars, so it has one balance. The mobile phone service tracks two balances, for dollars and minutes. The balance for minutes includes two sub-balances to track minutes granted under different conditions.

Figure 5-1 Balance Groups



The balance amount for a currency balance is normally either zero or a *debit balance*, which means the customer owes you. A debit sub-balance is represented as a positive number. The balance amount for a noncurrency balance such as minutes is normally zero or a *credit balance*, which means you owe the customer. A credit sub-balance is represented as a negative number.

For example, if a customer pays $30 per month for phone service and receives 100 minutes per month, the balances at the start of the month are:

- A currency balance of +30 US dollars
- A noncurrency sub-balance of -100 minutes

Noncurrency balances, such as minutes, are granted by charge offers and discount offers. For example, a customer can purchase an offer that grants 100 minutes every month. The number of minutes is recorded in the balance and then debited as the minutes are used.

Currency balances are not stored directly in balances. Instead, the amount shown in a currency balance is an amount drawn from several different bill items and accounts receivable items. For example, a balance amount of 50 dollars might be a combination of amounts in two different items:

- 60 dollars in usage fees, recorded in a usage billing item

- A 10 dollar credit recorded in an accounts receivable adjustment item

BRM tracks charges in these types of bill items:

- **Cycle forward items** track the accounts receivable for recurring cycle forward fees. See "About Subscription Charging" for information about recurring charges.
- **Cycle arrears items** track the accounts receivable for recurring cycle arrears fees.
- **Cycle forward arrears** items track the accounts receivable for recurring cycle forward arrears fees.
- **Usage items** track the accounts receivable for usage fees, purchase fees, and cancel fees. These fees are stored in the **/item/misc** object. See "How BRM Handles Data" for information about objects.

  An **/item/misc** object is created for each account and for every service that the account owns. This enables you to manage fees for each service independently; for example, you can display the usage fees for separate telephony services.
- **Custom items** track the accounts receivable for customized bill items you create.

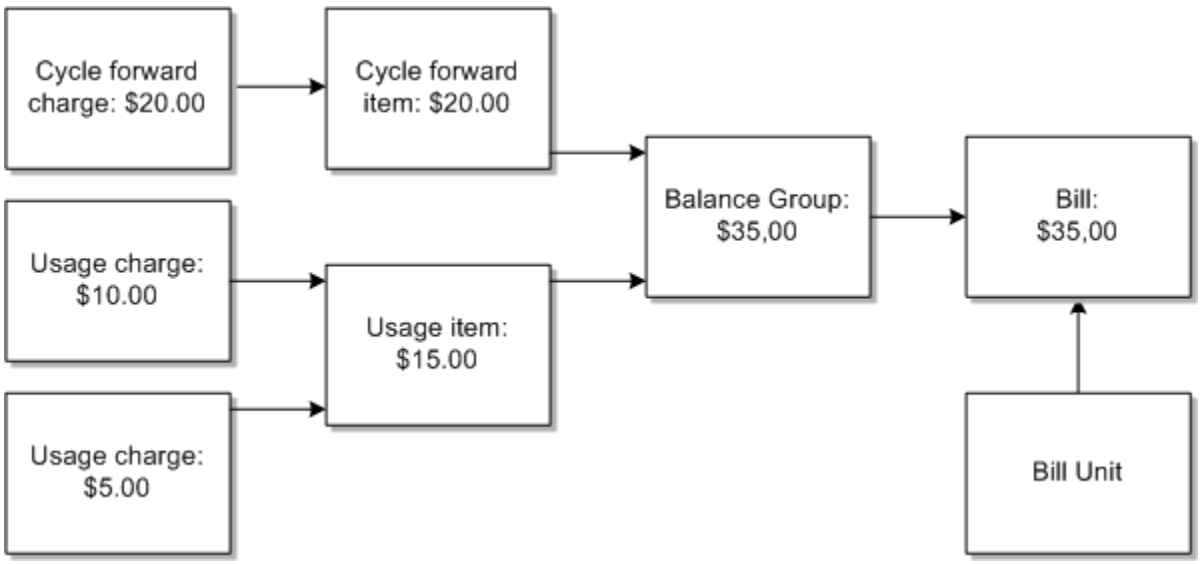
These bill items store the charges that accumulate in an account. When you run billing, the charges are consolidated into a bill. When the customer's payment is received, the balance amounts change to show that the charges have been paid for.

To bill an account, BRM must track and store the following information for each billing cycle:

- **The balance impacts for events.** This information is stored in bill items.
- **Total balance due from all events in a billing cycle.** The total balance is stored in a balance group.
- **When and how to generate a bill.** All administrative information for creating a bill, such as the payment method, billing cycle length, payment collection date, and hierarchy information, is stored in a bill unit.
- **The bill itself.** All information necessary to create a payment request is stored in a bill.

Figure 5-2 shows the role of bill items, balance groups, bills, and bill units in a typical billing cycle.

Figure 5-2 Balances In a Billing Cycle



About A/R Management

The money that your customers owe you before payments are received is your accounts receivable. You can perform several actions on a customer's accounts receivable:

- **Adjustments.** A CSR usually performs an adjustment to correct a problem. For example, a CSR might give an adjustment when your company charged the entire monthly fee for a service that was unavailable for a few days. A credit adjustment credits the currency balances in the customer's account and reduces the amount the customer owes. It does not return money directly to the customer as refunds.
- **Disputes.** A CSR creates a dispute when a customer disagrees with the amount he or she is asked to pay and the problem requires investigation before it can be resolved. A settlement is the resolution of a dispute; a settlement might favor the customer, favor the company, or divide the disputed amount between them. Disputes and settlements credit or debit the currency or noncurrency balances of a customer's account but do not return money to the customer directly.
- **Refunds.** You can give customers a refund whenever your company owes them money. Unlike an adjustment, which credits the customer's account balances, a refund returns money that your company owes a customer directly to the customer. Refunds for payments made by credit card or direct debit return money to the direct debit or credit card account the customer uses to make payments. For customers who pay invoices, your company makes a refund by check or other externally initiated payment method.
- **Write-offs.** A write-off removes from your company's assets an amount that your company has determined the customer will never pay. A write-off can also remove an amount that your company has decided it will not refund to the customer (for example, if the amount is very small or if you sent the customer a check that was returned because the customer moved without leaving a new address).

  You can reverse a write-off if the asset becomes available after it was written off.

To manage balances for these actions, BRM uses A/R items. A/R items include adjustment items, dispute items, settlement items, payment items, refund items, payment reversal items, write-off items, and write-off reversal items. A/R items collect balance impacts from the type of event reflected in their name.

> **Note:**
>
> To understand BRM accounts receivable, think of two types of items: bill items, which contribute to the total amount stored in a bill, and A/R items, which reflect actions on the A/R of an account or account bill, such as payments or adjustments.

Each A/R item tracks the following amounts:

- **Total**: The sum of all balance impacts to a currency balance.
- **Due**: The amount owed by the customer for this item. Initially, Due is the same as Total, but after A/R is received from or transferred to another item, the Total and Due may be different.

- **Adjusted**: The total amount of adjustments made to the item, the net of both debit and credit adjustments. Because most adjustments are credits (a negative number), the amount in the adjustment field usually reduces the Due of the item.
- **Disputed**: The total amount of unresolved disputes against the item. The disputed amount reduces the Due of the item.
- **Received**: The total amount transferred into this item from another item, which is usually a payment item.
- **Write-off**: A write-off is an A/R transaction that removes an uncollectable balance from a customer's account so it is not considered as an asset for accounting purposes.
- **Transferred**: The total amount transferred out of this item and into another item.
- **Status**: The state of the item, which can be Pending (unbilled), Open (billed), or Closed (zero amount due). Items generally move from Pending to Open to Closed status, but closed items can be reopened.

You can think of the Due, Adjusted, Disputed, Transferred, and Received fields as buckets that contain coins. As the BRM system manipulates A/R, BRM moves coins among these buckets, but it never changes the number of coins in the Total bucket. The Total bucket contains the coins from all the events linked to the item.

Figure 5-3 shows an example account's total charges in a cycle. The customer has accumulated $100 in charges. Without any adjustments, the customer has a balance due of $100.

Figure 5-3 Total Cycle Charges

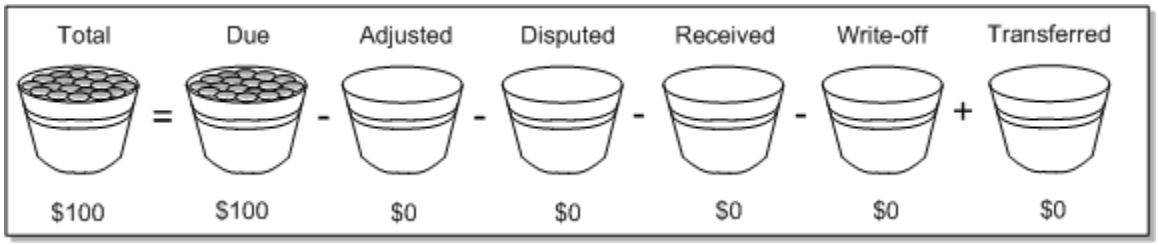
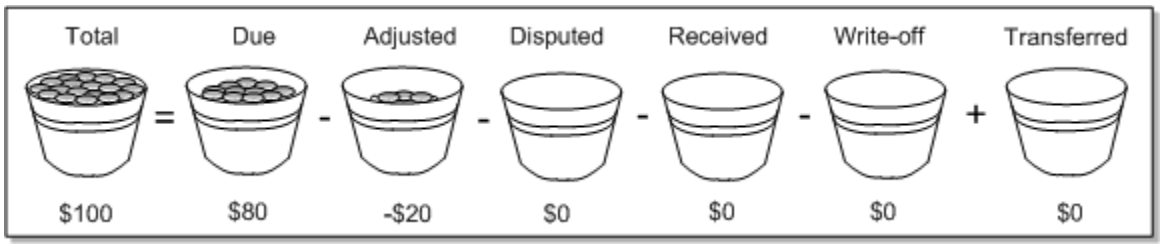


Figure 5-4 shows the same account holder's total charges of $100. An adjustment of $20 has reduced the customer's balance due to $80. The total value of the balance due and the adjustment still equals the total amount of charges.

Figure 5-4 Balance Due Impacted by Adjustment



Revenue Management Features

In addition to accounts receivable management, BRM includes the following revenue management features:

- **General ledger (G/L) reports.** Your company's general ledger is the total list of all revenue accounts used by your accountants to track revenue and expenses. For example, a company might use separate G/L accounts to track monthly fees and usage fees. To collect G/L data, you assign a G/L code to each type of balance impact when you create product offerings in PDC or Pricing Center. This results in a G/L amount for each charge. When you run a G/L report, BRM collects all of the revenue for each type of balance impact.
- **Tax calculation.** BRM can calculate taxes for charges. You can choose whether taxes are calculated using third-party tax software, such as Vertex, or by BRM. To calculate taxes, you apply a tax code to balance impact in product offerings.
- **Revenue assurance.** You use revenue assurance to verify the end-to-end completeness, accuracy, and integrity of BRM billing. You can analyze revenue assurance data to find revenue leakage in your system.
- **Collections.** Collections, or debt management, is a proactive process used by businesses to collect overdue payments from their customers. Collections includes identifying accounts that have overdue balances, determining whether those accounts meet predefined criteria for action, and then taking those actions.
- **Reports.** You can use BRM and Business Intelligence (BI) Publisher to run reports that show statistics about charges, service usage, and other revenue management data.

6

Overview of Payments

Learn about Oracle Communications Billing and Revenue Management (BRM) payments.

Topics in this document:

For information about setting up and running payment processing, see *BRM Configuring and Collecting Payments*.

About Payments

A payment consists of the amount and method by which customers pay their bills. There are several different payment methods available in BRM, depending on the type of payment processing your company performs.

Payments are typically requested from the customer at the end of the customer's billing cycle. For example, after billing is run, BRM can request a credit card payment or send an invoice.

When payments are received in BRM, a payment event is recorded, and the BRM system creates a payment accounts receivable (A/R) item. Payments can be submitted to BRM automatically, by a payment processor, or manually, by using Billing Care or Customer Center.

You use Billing Care or Customer Center to search for and review customer payments.

There are two types of payment processing in BRM: BRM-initiated and externally initiated. All payments received in BRM fall into one of these categories.

About BRM-Initiated Payment Processing

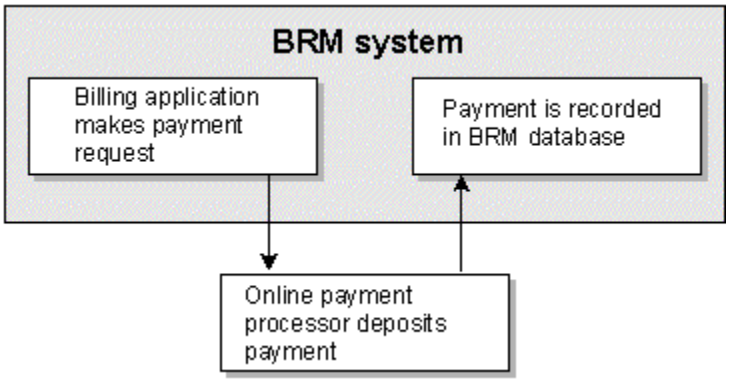
BRM-initiated payment processing is triggered by BRM and requires no action from customers. Payments processed in this way are those for which a customer is automatically charged, such as credit card and direct debit payments.

To begin processing such payments, BRM sends a customer's payment information to your online payment processor. The payment processing service charges the customer's credit card or checking account, and then BRM automatically allocates the payment and updates the customer's account balance. Any outstanding payment items are closed, and no payment management is required by a customer service representative (CSR).

You use the **pin_collect** utility to collect BRM-initiated payments. This utility is typically run automatically by one of the billing scripts. The customer's credit card is charged by your payment processor, and then the payments are sent to the BRM Payment Data Manager (DM).

Figure 6-1 shows how BRM handles BRM-initiated payments:

Figure 6-1 BRM-Initiated Payments



Customers who pay by credit card need credit card validation and authorization. *Credit card validation* validates the customer's address by checking the ZIP code and street address. *Credit card authorization* validates the customer's credit card by checking the card number, expiration date, credit limit, and so forth. By default, credit card validation occurs during account creation, and when a customer changes their credit card number. Credit card authorization occurs for every payment.

About Externally Initiated Payment Processing

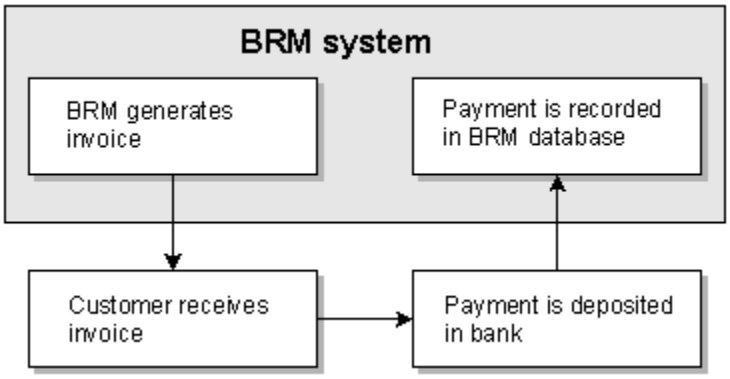
Externally initiated payments are typically cash or check payments made by the customer in response to an invoice.

Typically, after such payments are received from a customer, they are sent to a bank. The bank then initiates the payment processing by sending you a list of payments that have been received and deposited. If the bank sends the payment information through a directly integrated third-party service (payment gateway), BRM automatically allocates the payments and updates the customer's account balance. If the payment information is not sent through a payment gateway, you use Billing Care or Customer Center to allocate the payment and update the account.

You use Billing Care, Customer Center, or a third-party payment application to collect externally initiated payments and post them in BRM.

Figure 6-2 shows how BRM collects externally initiated payments:

Figure 6-2 BRM Collection of Externally Initiated Payments



You can submit the following externally initiated payments to BRM by using Billing Care or Customer Center:

- Check
- Cash
- Wire transfer
- Postal order
- Inter-bank transfer

About Payment Methods

A *payment method* is the mode by which customers pay their bills. The payment method is specified for an account when the account is created, but it can be changed at any time.

The following BRM-initiated payment methods are supported:

- Credit card
- Debit card
- Direct debit

> **Note:**
>
> Only debit cards that do not require a personal identification number (PIN) are supported in BRM.

The following externally initiated payment methods are supported in BRM:

- Cash
- Check
- Inter-bank transfer
- Postal order
- Wire transfer

- Voucher

In addition to those payment methods, BRM uses the following payment methods to manage payments in specific ways:

- **Invoice:** Accounts receive invoices on a monthly basis. These customers pay by check, cash, or other externally initiated payment methods.
- **Prepaid:** Accounts pay for service usage in advance. They send check or cash payments and can also pay by using a prepaid voucher.
- **Paid by parent account:** Accounts using this payment option are child accounts. These accounts have a nonpaying bill unit (**/billinfo** object) whose charges are paid by a paying bill unit in the parent account.
- **Undefined:** Accounts never receive a payment request. You typically use undefined accounts for free trial offers. Creating an undefined account enables a customer to create an account without needing to submit a credit card number.

About Payment Processors

You use payment processors to collect payments. The BRM system supports the following payment processors and middleware.:

- **Paymentech credit card and debit card processors.** You use the Paymentech DM.
- **Automated clearing houses (ACH).** An ACH is a secure system that connects banks with financial institutions and enables the electronic transfer of funds to and from checking and savings accounts. For example, you use an ACH to handle check payments and direct debit payments. ACHs do not handle credit card or debit card transactions.
- **Payment gateways.** A *payment gateway* is any third-party payment transaction application integrated with BRM. Payment gateways are designed by software development companies for enterprise merchants who already have a BRM billing system in place. A payment gateway provides a way for merchants to receive and process external payment data seamlessly with BRM. A payment gateway receives payment files from multiple sources (such as banks, credit card processors, and automated clearing houses), preprocesses them, formats the files for BRM, and calls BRM opcodes to record the payments in the database.

About Managing Payments

Payment processing includes the following activities:

- **Requesting payments from customers**. Payment requests are made by credit-card or direct debit (with BRM-initiated payments) or by sending the customer an invoice.
- **Receiving and validating payments.** Before payments can be processed by BRM, they must pass validation. Payment validation is initiated automatically (for example, by a credit-card processor) or manually by using Billing Care or Customer Center.
- **Allocating payments**. In most cases, payments are allocated automatically. For example, when a payment verification is received from a credit-card processor, BRM allocates the payment to the customer's bill unit. You can also allocate

payments manually (for example, allocate a partial payment to a specific bill item, or allocate payments to different bill units).

- **Reversing payments.** Payment reversals are necessary when a payment is recorded in the BRM database, but the payment is not deposited (for example, when a check does not clear). Reversing the payment enables BRM to treat the payment as if it never happened.
  Payments can be reversed manually by using Billing Care or Customer Center. Payment reversals also occur when payments are suspended and recycled.
- **Resolving failed payments**. When payments are received in BRM, they are posted as *successful* or *failed* by default. *Failed payments* are those that have been dishonored or rejected by the bank for financial reasons. For example, payments can fail due to expired credit cards, incorrect account details, or insufficient funds. You can use Payment Suspense Manager to manage failed payments.

Additional payment management features include the following:

- **Payment incentives.** You can provide payment incentives for customers who pay their bills early and in full. Incentives can affect currency balances, such as monetary credit (for example, a 5% reduction in the monthly bill amount), or noncurrency balances (for example, 20 minutes).
- **Payment fees.** You can charge a fee for failed payments.
- **Top-ups.** The BRM top-up features enable your customers to *top up* (add to) currency or noncurrency balances in their own accounts or in other customer accounts. For example, a customer can top up her account balance with a $50 payment from her credit card, or she can top up her teenage son's account with a $50 payment from her account. You can use the following types of top-ups:
  - **Standard top-ups:** Top-ups that customers make to their own accounts.
  - **Sponsored top-ups:** Top-ups made from one customer's account to another customer's account.
- **Payment channels.** The *payment channel* identifies how the payment was sent to your third-party payment processor, such as by the Web or Voice over IP (VOIP). You can create custom payment rules based on the payment channel ID (for example, you can grant a payment incentive if customers pay their bills over the Web).
- **Automatic payment collection.** By default, BRM-initiated payments, such as payments made by credit card or direct debit, are collected on the date that bills are finalized. You can configure BRM to collect a BRM-initiated payment on the date a bill is due or on a specified number of days before the bill is due.
- **Loans**. You can grant loans to prepaid customers when their account balance is insufficient for rating, or when their balance falls below a configured threshold. Any payments from top-ups or balance transfers are first used to pay off the loan.

7

Overview of Customer Management

Learn about Oracle Communications Billing and Revenue Management (BRM) customer management.

Topics in this document:

For information about creating accounts and managing customers, see *BRM Managing Customers*.

About Accounts

Every customer has an *account* in the BRM database. BRM accounts include or are linked to information about the customer's name and address, charge offers, discount offers, services, and balance.

In addition to customer accounts, BRM includes the following special-purpose accounts:

- The *root account* is used by system administrators to set permissions for customer service representatives (CSRs).

  > **Note:**
  >
  > BRM excludes the root account from billing. Therefore, BRM does not permit the root account to purchase product offerings.

- *CSR accounts* allow CSRs to log in to the BRM system. You also use CSR accounts to track CSR activity. See "Setting Up Permissions in BRM Applications" in *BRM System Administrator's Guide*.

BRM stores account data for each customer. For example:

- The customer's name and contact information.
- The account and service status (for example, Active or Inactive).
- The charge offers that the customer owns.
- The customer's balance.
- The customer's payment method (for example, invoice or credit card).

About Services

When a customer purchases a charge offer, BRM keeps a record of the charge offer's service information for the customer's account in the BRM database. For example, the email service includes information about the customer's login name and password, the maximum message size, and the maximum number of messages allowed in a mailbox.

An account can have multiple services, such as a phone service and an email service. An account can also have multiple services of the same type, such as multiple email accounts.

You can create customer accounts without charging for any services. For example, a customer can pay a monthly fee for an account with no services. In that case, a customer can add and cancel services and still be charged for having an account.

All services require a login name and password. For services such as telephony, where a customer does not use a login and password, login names and passwords can be generated automatically for internal use.

You control service usage in the following ways:

- Specifying how to authorize service usage. By default, a customer is not authorized to use a service if a credit limit has been reached.
- Inactivating services. When a service is inactivated, the customer cannot use it.

About Creating Customer Accounts

You can create customer accounts by using Billing Care, by using Customer Center, or by creating a custom application.

Account creation typically follows this process:

1. The customer calls the customer service representative (CSR) or accesses a Web page.
2. The customer chooses a package.
3. The customer provides information such as name, address, and credit card number.
4. The account is created in the BRM database. At account creation, a number of events occur:
    - The customer information is checked by BRM to see whether it is valid and to make sure that all required information has been supplied.
    - If the customer entered a credit card number, it is validated by a credit card processing service.
    - If there is a purchase fee, the customer is charged.
    - A welcome message is automatically sent to the customer.

Ways to Implement a Web Interface

You can implement a Web interface in the following ways:

- Use Self-Care Manager. You can customize the appearance of the default Web pages by modifying Java Server Pages (JSPs). A programmer can add functionality by modifying Self-Care Manager's source code.
- Create a Web interface by using the Portal Communications Module (PCM) library. If you use the PCM library, your Web server must run on a platform that supports BRM.

Adding Services to an Account

When a customer creates an account, they usually purchase charge offers that add services to their account. After the account is created, a customer can add services to their account.

To add services to an account, customers must purchase a package that includes the new service. A customer can purchase a bundle to add charge offers or discount offers for a service that they already own.

When a service is owned, you can change how the service is provisioned; for example, add mail boxes to an email account, or add IP telephony speed-dial settings.

About Activating, Inactivating, and Closing Accounts

An account's status can be active, inactive, or closed.

- When an account is *active*, the customer can use all active services.
- When an account is *inactive*, the customer cannot use any service. Inactivating an account prevents customers from generating usage and cycle balance impacts but does not prevent the account from being charged for bills already due.

  When you inactivate an account, you also inactivate all of its services, which prevents the customer from using the services. BRM does not charge usage and cycle fees while the account is inactive, but it does continue to collect bills that were due before the account was inactivated.
- When an account is *closed*, the customer cannot use any service. Closing an account cancels all of the charge offers and discount offers owned by the account.

  A closed account is normally closed permanently, but you can reactivate it. Closing an account does not delete the account or its service login names and passwords from the BRM database. Because accounts are never deleted from the database, there is no time limit on when they can be reactivated.

  When you close an account, all of the charge offers and discount offers owned by the account are canceled. When you re-activate the account, the offers must be repurchased.

> **Note:**
>
> You cannot delete accounts from a production BRM system.

You can customize how status changes are made. For example:

- You can enable customers with an inactive account to pay a bill by using a Web application.
- You can configure BRM to automatically inactivate or activate an account. You cannot automatically close an account or re-activate a closed account.

- You can backdate account status changes to a date earlier than the current date.
- You can schedule a status change for a future date.

About Service Status

Changing the status of an account also changes the status of all the account's services. Changing the status of a single *service* affects only that service.

- When you inactivate an account, all the account's services are inactivated.
- When you reactivate an inactive account, all the account's services that were inactivated when the account was inactivated are reactivated. Services that were inactivated independently of the account status are not reactivated.
- When you close an account, all the account's services are closed.
- When you reactivate a closed account, all the account's services that were closed when the account was closed are reactivated. Services that were manually inactivated (whose status was not changed when the account was closed) remain in the inactive state when the account is re-activated.
- You can specify a date in the future on which the service status will change.
- You can use service life cycles to add custom statuses. For example, you can add Dormant status to indicate a service that has not been used for a period of time.

About Charge Offer and Discount Offer Status

When creating an account or adding a bundle, you can change the status of each charge offer. For example, if using the offer requires hardware setup, you can inactivate the offer until setup is complete. The customer is not charged for the offer until it is activated.

Changing the status of an account also changes the status of all the account's charge offers and discount offers.

- When you inactivate an account, all the account's charge offers and discount offers are inactivated.
- When you reactivate an inactive account, you reactivate all charge offers and discount offers that were active when the account was active. Charge offers and discount offers that were inactive when the account was active remain inactive.
- When you close an account, all the account's charge offers and discount offers are canceled.
- When you reactivate a closed account, the account's charge offers and discount offers are *not* reactivated. To regain the canceled charge offers and discount offers, the account must repurchase them.

You cannot change the status of an offer after it has been purchased. However, you can change the start and end dates for offers to specify when balance impacts are recognized from the offer.

About Managing Purchased Offers

You can modify the offers that a customer owns in the following ways:

- Change the quantity that the customer can own (for example, add minutes to a balance).
- Change the validity dates (start and end dates) for purchase, recurring, and usage charges. Balance impacts are not recorded if the charges occur outside of the validity dates.
- Cancel the offer. You can cancel individual charge offers and discount offers, or cancel all offers in a bundle at one time by canceling the bundle. You can cancel an offer immediately or backdate the cancellation. You can also set the cancellation to a future date. You can charge for offer cancellations by creating a cancel charge.
- Configure bundle and package transitions. This enables you to define an offer upgrade path, and to enforce mutually exclusive offers.

You can configure several options for how to charge customers when canceling a charge offer or discount offer.

When Is Programming Required?

Most BRM features can be customized without any programming. For example, most product offerings, CSR-based account creation, and billing setup requires no programming to customize. The following customizations require some programming:

- **Creating services.** If you offer a service that is not supported by default in BRM, such as a fax service, you must create a custom service to capture and rate fax-related events.
- **Modifying some BRM defaults.** In some cases, you must modify policy source code to change BRM behavior. For example, to rate broadband access by the number of bytes downloaded rather than by connection time, you must edit source code.
- **Creating custom applications.** Some BRM implementations can be greatly enhanced by creating simple applications that manipulate data in the BRM database. For example, you can write applications to do the following tasks:
  - Alert customers that their free hours are almost used up.
  - Search for customers who have late payments.
- **Customizing Billing Care.** To change appearance and functionality, use the Billing Care SDK.
- **Customizing Customer Center.** To change appearance and functionality, use the Customer Center SDK.
- **Supporting a legacy system.** If you already have data stored in a database, you can write a Data Manager (DM) to provide an interface to that database.
- **Customizing event notification.** You use event notification to set up processes to run automatically. For example, you can customize BRM to send an email message to a customer automatically when a credit limit is nearly expired.
- **Implementing advanced pricing features.** Some pricing features (for example, product-level provisioning) require some programming for the initial setup.

About Customer Billing and Payment Information

You can manage the billing and payment information for each customer. Billing and payment information includes the following:

- Payment method (for example, if the customer pays by credit card or check).

- Credit card information.
- Invoice information (for example, the mailing address and whether to deliver the invoice by email or postal mail).
- The accounting type (balance forward or open item), which determines if bills are cumulative or not. See "About Accounting Types" for more information.
- The customer's billing day of month and billing frequency. See "About Billing Cycles" for more information.
- The customer's credit limit.

8

BRM System Architecture

Learn about the Oracle Communications Billing and Revenue Management (BRM) system components.

Topics in this document:

About the Four-Tier Architecture

The BRM system uses a four-tier architecture consisting of the following tiers:

- Application Tier
- Business Process Tier
- Data Management Tier
- Data Tier

Figure 8-1 shows the BRM system architecture:

Figure 8-1 BRM System Architecture

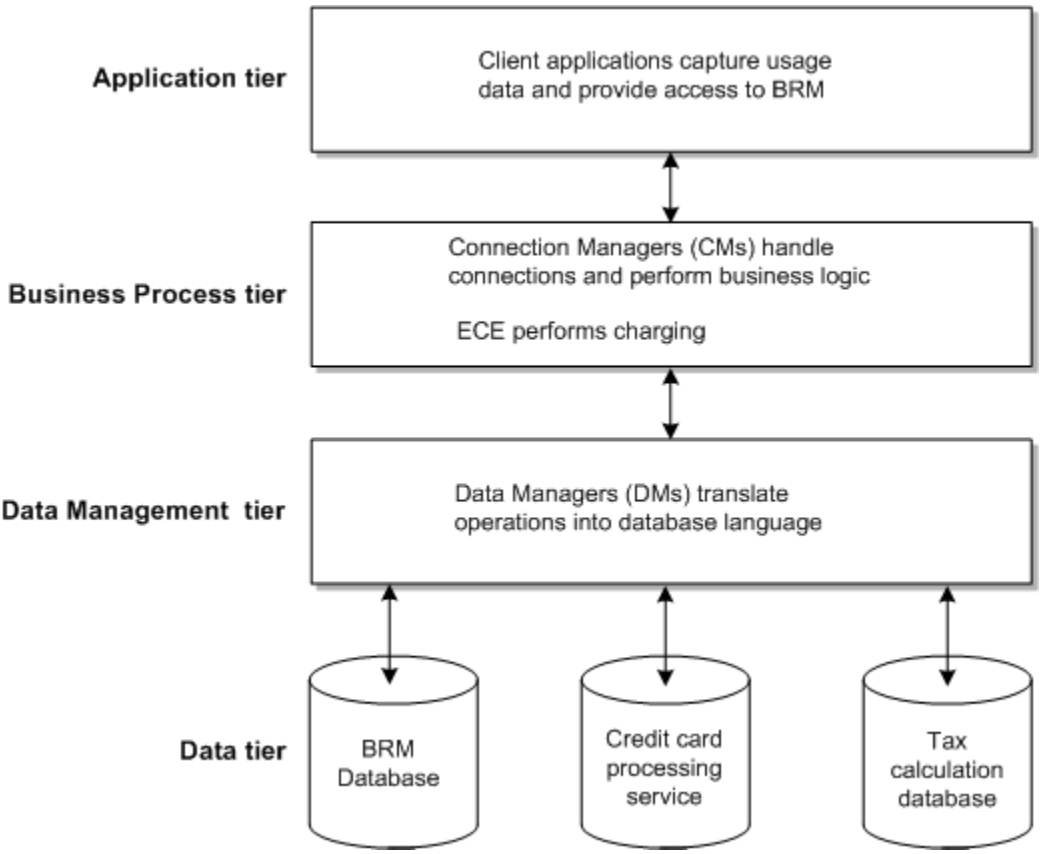
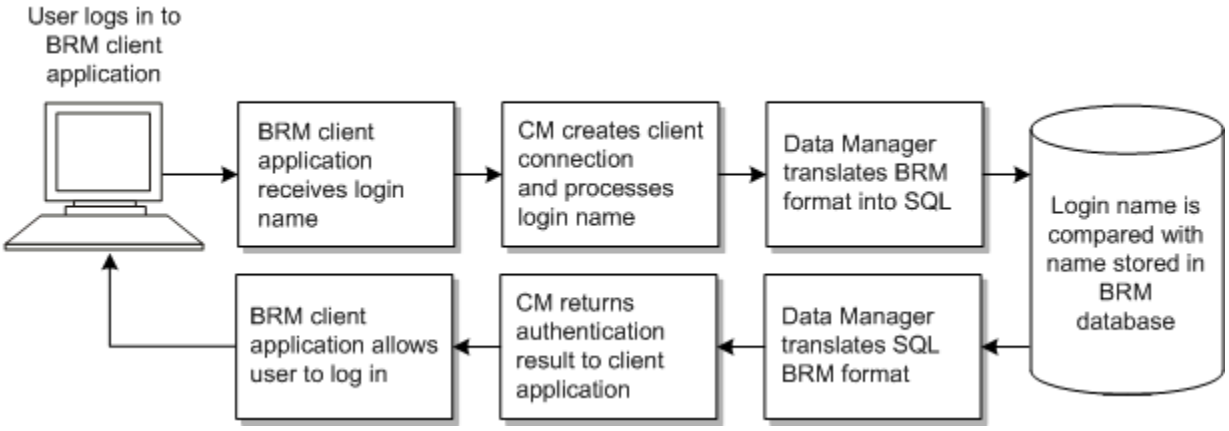


Figure 8-2 shows how the BRM components process a login name when a user starts a BRM client application.

Figure 8-2 BRM Customer Login Process



Application Tier

The application tier consists of the following types of client applications:

- BRM client applications used to create accounts and manage customers and their accounts, such as Billing Care or Customer Center
- Third-party customer account management applications used by customer service representatives (CSRs), such as applications that capture data from customer service usage
- BRM service integration applications such as Global System for Mobile Communications (GSM) Manager and General Packet Radio Service (GPRS) Manager
- Network connection clients; such as Offline Mediation Controller
- Custom applications that you write to manage customers or support custom services and integrate your applications with BRM

Business Process Tier

The business process tier is responsible for running BRM functionality, such as the following:

- Online and offline charging, discounting, and promotions
- Customer and subscription management
- Real-time balance and credit management
- Billing
- Service usage authorization
- Payments and A/R management

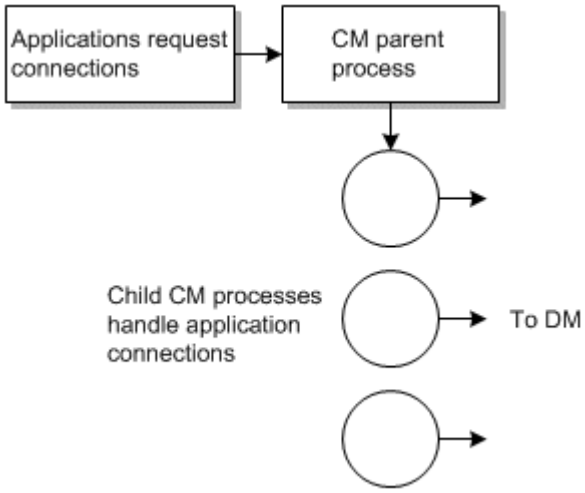
The business process tier consists of the following components:

- **Connection Managers (CMs):** CMs provide an interface between clients and the rest of the system. All client applications connect to the BRM system through a CM.
- **Facilities Modules (FMs):** CMs run FMs that process the data captured by the client. For example, when a user logs in, FMs process the login name and password. See "About Facilities Modules (FMs)".
- **External Modules (EMs):** EMs serve similar functions to FMs but must be started separately as a service or process. See "About External Modules (EMs)".
- **Elastic Charging Engine (ECE):** ECE performs online charging and offline charging. During online charging, ECE performs real-time authorization and balance management.

  If you are using Pipeline Manager for rating and discounting events in batch and real-time, see "About Pipeline Rating" in *BRM Configuring Pipeline Rating and Discounting*.

CMs run as daemons. When a client application requests a connection, the parent CM process spawns a child process to handle the connection. At that point, the application no longer communicates with the parent CM; all communication flows from the application to the child CM as depicted in Figure 8-3.

Figure 8-3 Client Connections to Child CM



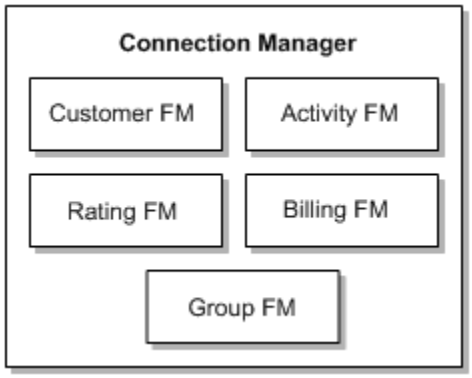
About Facilities Modules (FMs)

When an application connects to a CM, the data sent by the application is processed first by the FMs that are included in the CM.

There are separate FMs to handle different types of tasks. For example, if a CSR changes a customer's password, the Customer FM validates that the new password has the correct number of characters.

FMs manage BRM activity and ensure that data is processed correctly. When you configure a CM, you specify the FMs that are linked to the CM. Typically, you use the default set of required FMs for each CM. Figure 8-4 depicts a CM configured with several FMs.

Figure 8-4 Facilities Modules for CM



Each FM is dynamically linked to a CM when the CM starts. You can link optional or custom FMs to any CM.

Because FMs are linked to CMs, the FMs can get configuration information from the CM configuration files. Therefore, one of the ways you can customize BRM policies is by editing the CM configuration file.

You can customize policy FMs or add custom FMs to a CM. See *BRM Developer's Guide*.

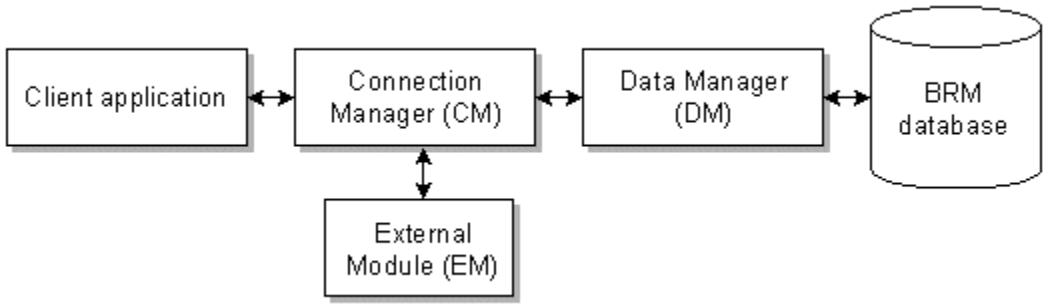
About External Modules (EMs)

An External Module (EM) is similar to an FM; it is a set of opcodes that perform BRM functions. However, it is not linked to a CM in the same way that an FM is. Instead, it runs as a separate process that you must start and stop.

The Payload Generator is an EM that processes events that are exported to external systems by using the optional Enterprise Application Integration (EAI) framework.

Figure 8-5 shows where an EM fits in the system architecture:

Figure 8-5 External Module in BRM System Architecture



About Connection Manager Master Processes (CMMPs)

Instead of configuring an application to connect directly to a CM, you can configure it to connect to a Connection Manager Master Process (CMMP).

When an application connects to a CMMP, the CMMP selects a CM from a list you provide and gives the application the machine name and port of that CM. The application then uses that address to connect to the CM.

Figure 8-6 shows how an application connects to a CM using a CMMP:

Figure 8-6 Using a CMMP to Connect to a CM

For more information, see the information about improving performance in *BRM System Administrator's Guide*.

Data Management Tier

The data management tier consists of Data Managers (DMs), which translate requests from CMs into a language that the database can understand. For the BRM database, the language is SQL. There are also DMs for credit card processors, Vertex, and so forth. See "About Data Managers (DMs)".

About Data Managers (DMs)

Data Managers (DMs) translate BRM operations into a form that can be understood by the data access system.

BRM includes the following DMs:

- The Oracle DM (**dm_oracle**) provides an interface to an Oracle BRM database.
- The Paymentech DM (**dm_fusa**) provides an interface to the Paymentech credit-card payment processor.
- The Vertex DM (**dm_vertex**) provides an interface to the Vertex Sales Tax Q Series and Communications Tax Q Series database.

Child processes are not spawned when a DM receives a connection request. Instead, child processes are spawned when a DM starts. You define the number of child processes in the DM configuration file.

When a DM receives a connection from a CM, the parent process or thread assigns a child process or thread to the connection. At that point, the client no longer communicates with the parent DM, and all communication flows from the CM to the child DM.

DMs run as a set of processes. Each process handles a single database transaction.

About Data Manager (DM) Back Ends

Each DM has a set of front-end processes that receive connection requests and send them to a shared-memory queue for processing by back-end processes. The back-end processes translate the transaction into the database language.

DM back ends are single-threaded. Therefore, a back end is dedicated to a single transaction for the duration of the transaction.

From the BRM database perspective, the back ends communicate to database back-end (shadow) processes. One database back-end process is started for each DM back end. See your database documentation for information on shadow processes.

You can write a new DM back end to integrate BRM with any relational, object-oriented, or legacy database management system. You can also layer BRM on top of basic file managers. For more information, see the documentation about writing a custom Data Manager in *BRM Developer's Guide*.

Data Tier

The data tier consists of the BRM database and other data access systems, such as the Paymentech credit-card processing service.

About the BRM Database

The BRM database stores all of your business information and data, such as information about customer accounts, your product offerings, and the services you provide. All information stored in the BRM database is contained in objects.

An *object* is equivalent to a database record or a set of database records. Each type of object contains a set of related information. For example, an account object includes the customer's name and address.

Related database objects are linked to each other. For example, an account object is linked to the payment information object that stores a customer's credit card number.

To see examples of objects, use Object Browser.

> **Note:**
>
> The term *object* does not refer to an object request broker (ORB) or to the object programming environment. Instead, *object* refers to the BRM object data model. For more information about the object model, see *BRM Developer's Guide*.

About the Residency Type

RESIDENCY_TYPE is an attribute of an object that defines where the object resides in a BRM system. Residency type values are predefined in the data dictionary in the BRM database.

Table 8-1 lists the BRM residency types:

Table 8-1 Residency Type Descriptions

RESIDENCY_TYPE Value	Object Type	Description
0	Database object	Resides in the BRM database.
9	Global database view object	Resides in the data dictionary. Use with database views to perform distinct searches on multiple storable classes using pagination and ordering. See "Performing Distinct Searches with Ordering and Pagination" in *BRM Developer's Guide*.
101	Routing database object	A **/uniqueness** object that resides in the primary schema of the BRM database.

If you create a custom object, you must set its residency type to the correct value.

About the Multischema Architecture

In multischema systems, the database layer of your BRM system consists of one primary schema and one or more secondary schemas in a *single database*. A primary DM is connected to the primary schema and secondary DMs are connected to the secondary schemas. Data is separated between the schemas as follows:

- The *primary database schema* stores two types of global objects: objects that the secondary DMs can only read, such as configuration and pricing objects, and objects that the secondary DMs can both read and update, such as uniqueness objects. The primary schema also stores subscriber data.
- The *secondary database schemas* store subscriber data, such as event and account objects.

The primary DM updates global read-only objects in the primary schema, and the secondary DMs read the data from views in the secondary schema on tables from the primary schema.

The secondary DMs use schema qualifications to read and modify updatable global objects stored in the primary schema.

For information about how to set up a multischema system, see "Managing a Multischema System" in *BRM System Administrator's Guide*.

About Multidatabase Manager

Multidatabase Manager is an optional feature that enables you to have multiple BRM database schemas in a single installation. This option can support very large installations (more than a few million subscribers). It enables you to split the main database into multiple schemas.

About Oracle RAC for a High-Availability BRM System

For a high-availability system, BRM recommends Oracle Real Application Cluster (Oracle RAC), which requires a reliable and highly available storage system. For more information on a high-availability system, see *BRM System Administrator's Guide*.

Configuring the Four-Tier Architecture

All BRM processes can run on the same computer, or they can be distributed among different computers. Distributed processing provides a lot of flexibility in configuring your system, especially if you have multiple machines running CMs and DMs. For example, you can do the following:

- Run redundant system processes (for example, Billing Care, billing applications, CMs, and DMs).
- Connect clients to specific CMs, or use a CMMP to route to any available CM.

For more information, see *BRM Installation Guide*.

Communication between System Components

Communication between applications and CMs, and CMs and DMs, is accomplished by TCP/IP. You assign each component an IP port and host name. TCP/IP facilitates the use of firewalls, proxies, and filters.

Any kind of network connection that supports TCP/IP supports BRM (for example, local area network, virtual private network, and PPP).

Communication between DMs and data access systems uses the protocol required by the data access system. For example:

- DM-to-Oracle communication is carried out using Oracle SQL*Net, a product that lets distributed computers access a central Oracle database.
- DM-to-Paymentech communication uses the Paymentech protocol.

For information about the configuration file entries that enable communication between BRM components, see the documentation about configuration files in *BRM System Administrator's Guide*.

Four-Tier Architecture and Failure Recovery

The four-tier architecture provides high reliability in the event of system component failures:

- If the BRM database is offline, any interrupted transactions are rolled back when the database is restarted. The DM automatically attempts to connect to the database until a connection is made.
- If a DM is offline, any interrupted transactions are timed out and rolled back by the database. The CM automatically attempts to connect to the DM until a connection is made. You can provide each CM with a list of DMs. If a DM is unavailable, the CM can connect to a different one.

  If a DM fails while there are transactions in its queue, the transactions must be resubmitted. The contents of the queue are lost when the DM fails.
- If a CM is offline, the database rolls back any interrupted transactions. You can use a CMMP to provide multiple CMs for clients to connect to. See "About Connection Manager Master Processes (CMMPs)".
- If a client fails, any interrupted transaction is timed out and rolled back by the database.

  In all cases, errors are reported to the client application. Depending on its capabilities, the client can display an error, log the error, or retry the transaction.
- In all cases, broken transactions are rolled back by the database, so no partial transactions are recorded.

ECE System Architecture

ECE can be seen as a separate sub-system with the overall BRM system. ECE includes its own components, and connects to most of the other BRM components, including PDC and the BRM server.

The Elastic Charging Server runs all of the charging, balance query, and policy control functions. The Elastic Charging Server runs on Oracle Coherence, which distributes

processing across multiple processing nodes, ensuring high availability and scalability (giving the server *elastic* properties). ECE nodes perform multiple functions; for example, *charging nodes* run the charging logic, and *cache nodes* store the customer data and product offering data required for charging.

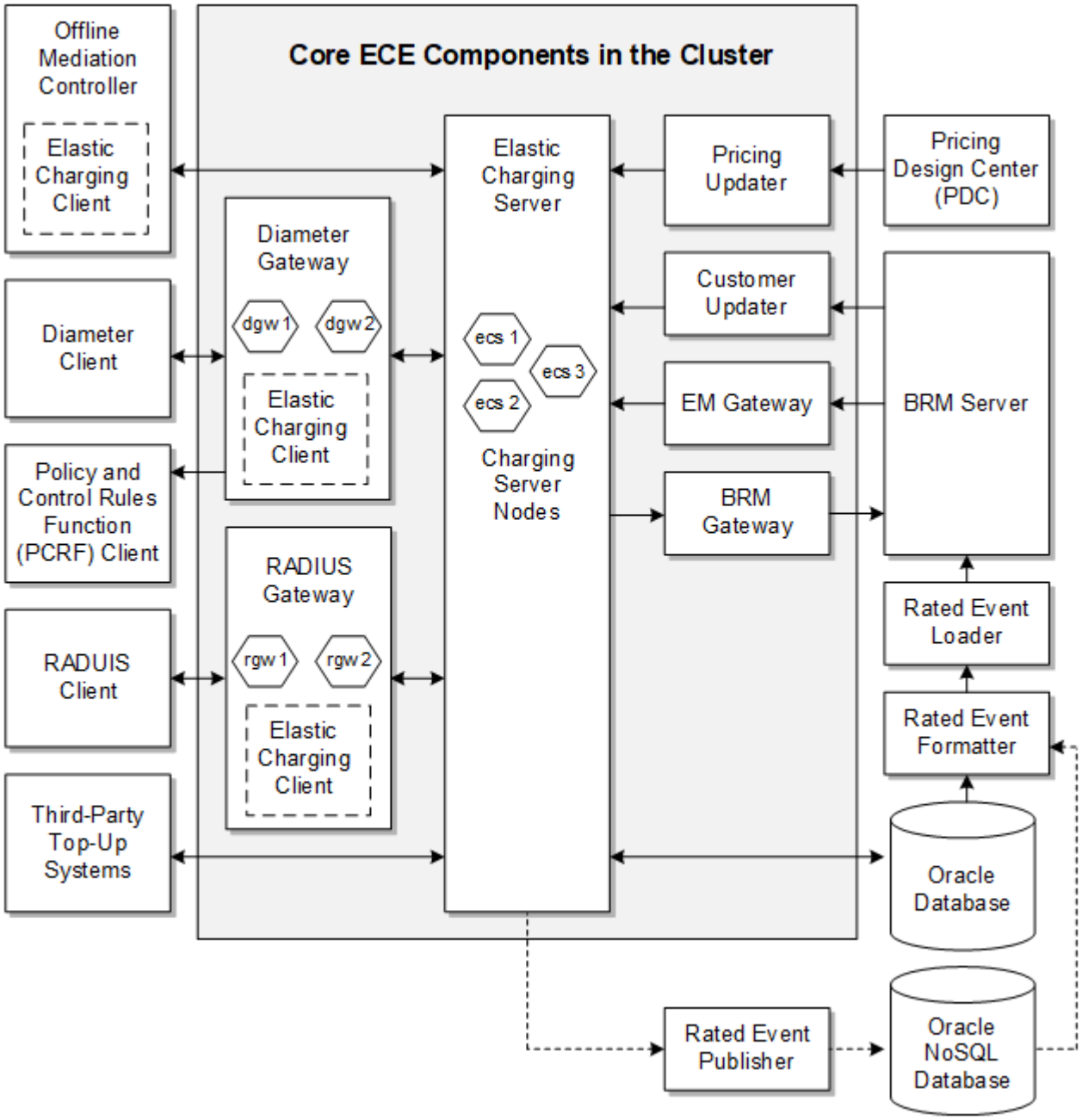
The ECE system architecture can be divided generally into these functional areas:

- Communicating with the network elements that ECE receives events from.
- Communicating with other BRM components (BRM server and PDC) to synchronize customer data and product offerings.
- Sending rated events to the BRM database.

Figure 8-7 shows the ECE system architecture:

A dotted line depicts the data flow for storing rated events using Oracle NoSQL Database (only if data persistence is disabled).

Figure 8-7 ECE system Architecture



To communicate with the network elements, ECE uses these components:

- **Elastic Charging Clients** provide the interface between network elements and the ECE server. For example, Offline Mediation Controller uses an Elastic Charging Client to send offline rating usage requests to the Elastic Charging Server.
- **Diameter Gateway** and **RADIUS Gateway** provide integrations to Diameter and RADIUS clients. In both cases, the gateways handle requests and responses between the network elements and ECE. Both gateways use an Elastic Charging Client to manage communication with the Elastic Charging Server.

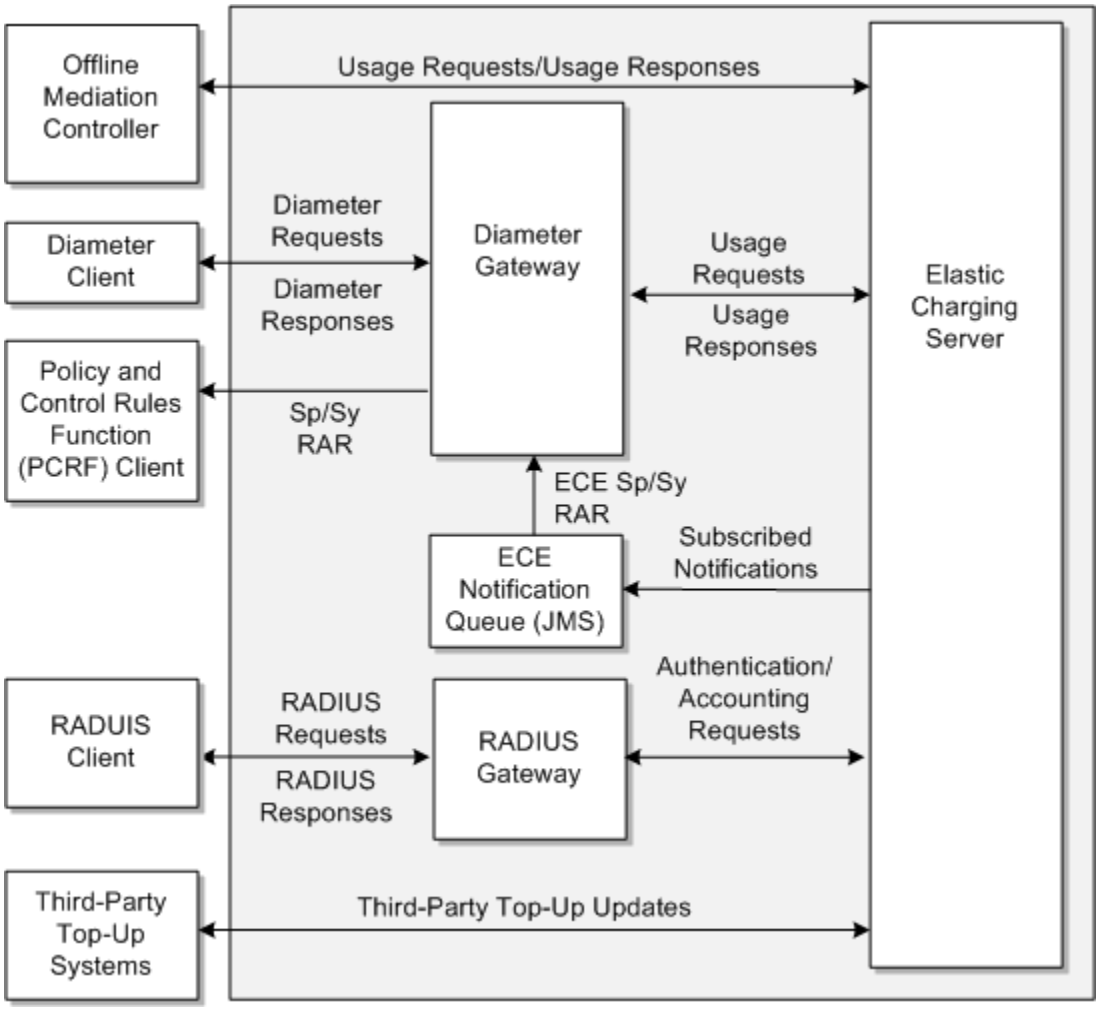
  - Diameter Gateway handles usage requests and usage responses.
  - RADIUS gateway manages authentication and accounting requests.

  Diameter Gateway reads from the ECE notification queue (JMS topic) and processes push notifications from Elastic Charging Server. From the ECE push notifications (ECE re-authorization request (RAR) notifications, ECE spending limit notifications, and ECE subscriber preferences notifications), Diameter Gateway constructs Sy and Gy messages and sends them back to Diameter clients (for example, for answering back to push-notification-request (PNR), subscribe-notification-request (SNR), and RAR notifications).

- Diameter Gateway also communicates with PCRF policy clients. When integrating PCRF policy clients with ECE, ECE acts as the Subscriber Profile Repository (SPR) because it stores the customer profile information used by the PCRF. ECE offers a combined Sp and Sy interface which the PCRF uses to retrieve customer preferences and policy counter information.
- **Top-up clients** use the ECE API to communicate directly with the Elastic Charging Server.

Figure 8-8 shows the communication that occurs between the network elements and the Elastic Charging Server.

Figure 8-8 Communication Between ECE and the Network



To integrate with the BRM server and PDC, ECE uses the following components:

- **Pricing Updater** listens on a JMS queue and receives updates to product offerings from PDC. It then loads the updated product offering data into an ECE cache whenever you change product offering data in PDC.

  > **Caution:**
  >
  > When configuring a development system, you can use the **pricingLoader** utility to load sample product offerings. However, do not use **pricingLoader** utility on a production system, or on a test system that includes Pricing Updater.

- **Customer Updater** receives customer data from the BRM server. Customer Updater does the following:
  - Performs the initial loading of customer data, credit limit data, offer profiles, product offering cross-reference data, and configuration objects, from BRM and loads the data into ECE.

    Product offering cross-reference data is stored in ECE and enables ECE to map the pricing components stored in the BRM database to the pricing components stored in ECE. ECE maps pricing components by using the pricing component POIDs. The product offering cross-reference data is initially loaded into ECE by Customer Updater. Subsequent new or modified pricing components are loaded into or updated in ECE through EM Gateway.
  - Handles asynchronous updates from BRM to ECE of the following data: rerating, account migration, discount, and product.

  When events occur that update rerating, account migration, discount, and product data in the BRM database, the updates are published asynchronously (not in real time) to ECE through Customer Updater. To do so, the updates are sent in business events to the Oracle DM, which publishes the business events to an Oracle AQ database queue. Customer Updater retrieves the business events from the queue and updates the ECE cache accordingly.
- The **EM Gateway** receives customer data from the BRM server in real time. The EM Gateway does the following:
  - Loads customer data from the BRM server into an ECE cache in real time.
  - The BRM rerating utility uses the EM Gateway to send *rerating requests* from BRM to ECE during the rerating process.
  - Custom BRM applications can use the EM Gateway to send balance query requests to ECE.

  Customer Updater and the EM Gateway both receive updates to customer data from BRM server and load it into the ECE customer data cache. However, they load data differently.

  The EM Gateway loads customer data in real time. When an event occurs that changes customer data in the BRM database:

  1. A transaction is opened to update the BRM database.
  2. Before the transaction can be completed, the BRM server sends the updated data to ECE by using the EM Gateway.
  3. ECE updates the data and returns a message to the BRM server.

4. The transaction is completed in the BRM database. If ECE is not able to update the data, the transaction is rolled back.

   This process ensures that the data is synchronized in ECE and in the BRM database.

- The **BRM Gateway** sends information to the BRM server when processes must be triggered in the BRM server, or data must be updated in the BRM database. For example, if auto-triggered billing is enabled, ECE uses the BRM Gateway to trigger billing in the BRM server.

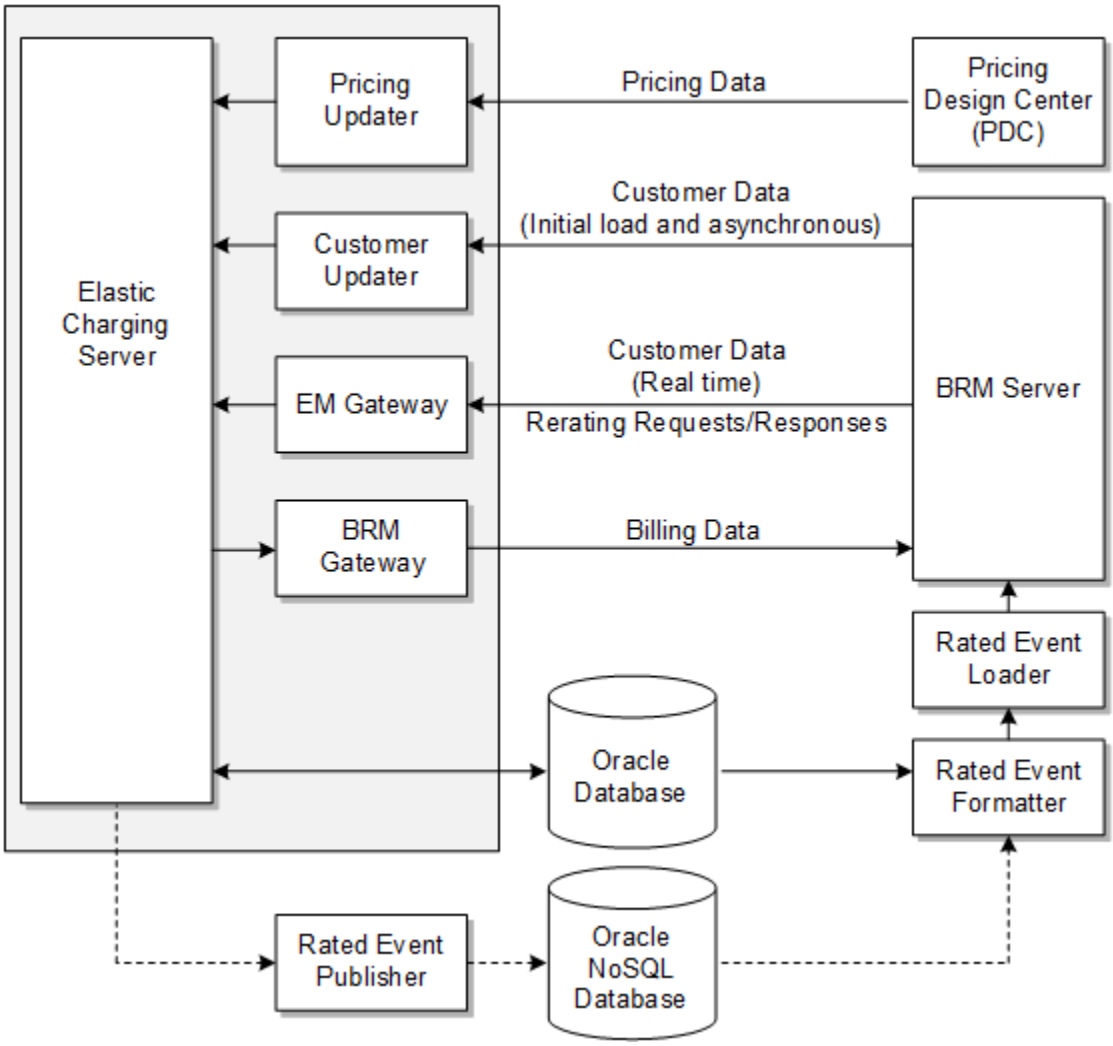
  BRM Gateway connects to the CM to make BRM opcode calls. The CM calls opcodes implemented in an external manager (EM). External Manager (EM) Gateway receives requests from the CM at a predefined port.

If data persistence is enabled in ECE, all the data received from BRM and PDC are persisted in an Oracle database by default. When you restart the components, the data in this database are reloaded into the ECE cache.

Figure 8-9 shows the communication that occurs between the Elastic Charging Server, BRM server, and PDC:

A dotted line depicts the flow for sending rated events to BRM using Oracle NoSQL Database.

Figure 8-9 Communication Between Elastic Charging Server, BRM Server, and PDC



The components that send rated events to the BRM database are **Rated Event Publisher**, **Rated Event Formatter**, and **Rated Event Loader**. These components load rated events from the Elastic Charging Server into the BRM database.

If data persistence is enabled, ECE stores the rated events into an Oracle database. Rated Event Formatter extracts the rated events from the Oracle database and formats the event data into a format that can be loaded into the BRM database. Rated Event Loader loads the events into the BRM database.

If data persistence is disabled, Rated Event Publisher publishes rated events from the Elastic Charging Server to an Oracle NoSQL database. Rated Event Formatter extracts the rated events from the Oracle NoSQL database and formats the event data. Rated Event Loader loads the events into the BRM database.

The Oracle NoSQL database deletes duplicate rated events. For example, when an ECE charging server node fails after having sent rated events to the NoSQL database, the node will resend the rated events to the NoSQL database upon startup. Oracle NoSQL Database overrides the existing data to prevent duplicate data entries.

To configure and manage the ECE system, you perform two basic types of tasks:

- Connect the ECE server with external clients and systems, such as Offline Mediation Controller, the BRM Gateway, and the EM Gateway. These connections are managed by JMS queues, and are configured during the ECE installation. You can change them after installation to tune performance.
- Manage the running ECE system. To manage ECE, you configure a driver machine. The driver machine stores the configuration files that define the ECE system. You log in to the driver machine to run ECC.

  Managing the ECE system includes managing nodes, such as adding charging server nodes that perform rating and store data. Running the ECE software and storing data on multiple nodes provides high availability for usage charging.

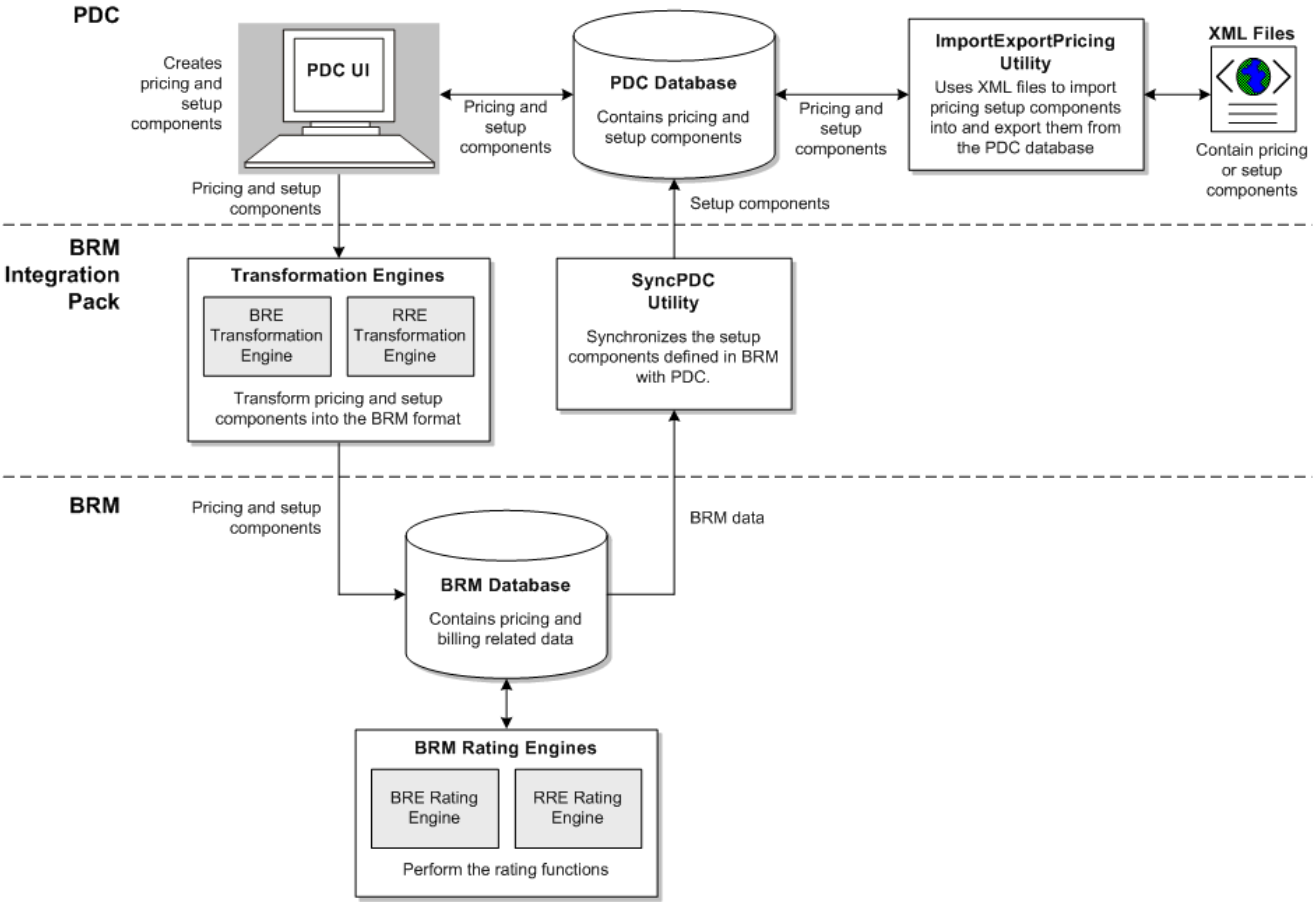
PDC System Architecture

The PDC application architecture consists of the following components:

- **PDC UI**. The PDC UI is a Web-based user interface that you use to create and manage your product offerings.
- **PDC database**. The PDC database stores the pricing components and setup components that you configure in PDC, as well as the pricing data defined in Oracle Communications Billing and Revenue Management (BRM) and rating systems that are synchronized with PDC.
- **BRM Integration Pack**. The BRM Integration Pack provides the interface between PDC and BRM.
- **ImportExportPricing utility**. The **ImportExportPricing** utility provides an XML-based command-line interface for importing pricing and setup components to and exporting pricing and setup components from PDC.

Figure 8-10 shows the PDC application architecture and the process flow when used with BRM:

Figure 8-10 PDC and BRM Integration Architecture



About the PDC and BRM Integration Process Flow

The BRM Integration Pack provides the interface between PDC and BRM to transform and load the pricing data configured in PDC into BRM and to synchronize the pricing data defined in BRM and the rating system with PDC.

The BRM Integration Pack contains:

- **Transformation engines**: The transformation engines convert pricing components and setup components configured in PDC into the format required by BRM and then loads the components into the BRM database.
- **SyncPDC utility**: The **SyncPDC** utility provides the capability to synchronize setup components defined in BRM with PDC. It also synchronizes event data defined in the rating system with PDC.

BRM uses the pricing and other data to measure customer activity and determine how much to charge the customer for the use of the services. See BRM documentation for information on how the rating is performed.

9

About Implementing Billing and Revenue Management

Learn how to implement Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

Implementation Process

A typical BRM implementation process includes these tasks:

1. Configuring the BRM definition of services and events. For example, if you provide a service that is not configured in BRM, you need to define the ***/service*** and ***/event*** classes to support that service. See *BRM Developer's Guide*.
2. Creating your product offerings.
3. Setting up customer account creation.
4. Configuring billing, payments, and other revenue management processes.
5. Testing your implementation on a test system.
6. Installing a production system.

Ways to Use and Customize BRM

You can customize BRM to change the default BRM business functionality. By customizing BRM, you can change how BRM responds to customer activity. For example, you can do the following:

- Change the net due date for invoice accounts.
- Specify the reasons for automatically inactivating an account.
- Specify how and when to validate credit cards.
- Collect additional data about a customer (for example, the customer's bank ABA number, employee number, or member referral information).

You customize business functionality by editing configuration files and by customizing BRM source code. You do not need to be a programmer to edit configuration files, but you must be a programmer to customize source code.

You can use several methods to use and customize BRM features:

- Use BRM client applications, such as Billing Care, to manage customers, record payments, and change BRM default behavior. Client applications run on the Windows platform and have a graphical user interface.
- Use BRM command-line utilities such as the **pin_collect** utility and the **load_pin_glid** utility to run billing utilities and load data into the BRM database.
- Edit configuration files to change the default BRM behavior and tune system performance.
- Customize BRM source code. This requires programming.

> **Caution:**
>
> – Always use the BRM API to manipulate data. Changing data in the database without using the API can corrupt the data.
>
> – Do not use SQL commands to change data in the database. Always use the API.

About Creating Custom Account Management and Billing Tools

Using the BRM APIs, you can create custom client applications and customer interfaces. For example:

- Web-based applications that customers use to change their account status, upgrade their services, and pay their bills.
- An online account creation package requiring no existing broadband access and distributed on a variety of shareware CDs.
- An application that runs nightly, searches for accounts that have a free trial account, and sends email when the free trial account is about to expire.
- An application that runs nightly to check the age of an account. If the account is over two years old, the customer is automatically upgraded to a discounted package.
- An application that checks customer profiles and applies a discounted charge for teachers.
- An application that monitors the customer's disk space on a server and charges more if allotted disk space is exceeded.

About Connecting BRM to External Data Processing Sources

BRM is modular and extensible by design. Many customers add custom interfaces to external components. Some customers use external components instead of standard BRM components. For example, some customers use BRM rating but use their own proprietary billing system. Other customers use a third-party customer management system.

Examples of external integrations:

- Tax calculation services.
- Credit card processing services.
- Legacy billing systems.
- Customer management systems.
- Credit checking services.
- Accounting applications to handle A/R and G/L data.
- Business information distribution services (for example, services that automatically distribute the results of database reports).
- Fulfillment services that send welcome letters and goods such as CDs.
- Invoice processing, printing, and mailing companies.
- Offsite customer service organizations.

About the Data Displayed in BRM Client Applications

You can change some information displayed in client applications to support your business needs. For example:

- Change the list of packages or edit your package lists.
- Change the predefined reasons that describe why a change was made to an account.
- Add a payment method.
- Change the list of currencies you support.

About BRM Defaults

Almost all business policies have a default implementation. For example, the default billing cycle is one month. You can change the default implementation by editing configuration files, or customizing policy code.

When Is Programming Required?

Most BRM features can be customized without any programming. For example, most product offerings, CSR-based account creation, and billing setup requires no programming to customize. The following customizations require some programming:

- **Creating services.** If you offer a service that is not supported by default in BRM, such as a fax service, you must create a custom service to capture and rate fax-related events.

- **Modifying some BRM defaults.** In some cases, you must modify policy source code to change BRM behavior. For example, to rate broadband access by the number of bytes downloaded rather than by connection time, you must edit source code.
- **Creating custom applications.** Some BRM implementations can be greatly enhanced by creating simple applications that manipulate data in the BRM database. For example, you can write applications to do the following tasks:
  - Alert customers that their free hours are almost used up.
  - Search for customers who have late payments.
- **Customizing Billing Care.** To change appearance and functionality, use the Billing Care SDK.
- **Customizing Customer Center.** To change appearance and functionality, use the Customer Center SDK.
- **Supporting a legacy system.** If you already have data stored in a database, you can write a Data Manager (DM) to provide an interface to that database.
- **Customizing event notification.** You use event notification to set up processes to run automatically. For example, you can customize BRM to send an email message to a customer automatically when a credit limit is nearly expired.
- **Implementing advanced pricing features.** Some pricing features (for example, product-level provisioning) require some programming for the initial setup.

10

About Configuring BRM

Learn how to configure Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

Ways to Configure BRM

The main types of configurations in BRM are:

- Configuring system components; for example, connecting a Connection Manager (CM) to a Data Manager (DM) by configuring a host name and port number.
- Configuring business logic options; for example, specifying the default billing day of month.
- Loading data used by BRM; for example, general ledger IDs.

The primary ways to configure BRM are:

- Using GUI applications.
- Editing configuration files, such as **pin.conf** files and **Infranet.properties** files. See "About Editing pin.conf Files" and "About Editing an Infranet.properties File".
- Editing business parameters. See "About Editing Business Parameters".
- Editing and loading data files; for example, a file that defines tax calculation data. See "Loading Configuration Data Into the BRM Database".

About Editing pin.conf Files

Use **pin.conf** files to configure connections between system components, and to configure business logic operations, such as how billing works. For example, you can edit a **pin.conf** file to specify the default billing day of month.

pin.conf files are text files, and can be edited by using any text editor. Each entry in a **pin.conf** file includes instructions on how to configure it.

A typical **pin.conf** entry looks like this:

```
- fm_bill allow_move_close_acct 1
```

In this example, **1** enables the entry, and **0** disables it.

Note the following points when editing **pin.conf** files:

- **pin.conf** files are usually stored in the same directory as the utility or component that uses them. For example, the **pin.conf** file that you edit to configure a Connection Manager is stored in *BRM_home***/sys/cm**.

  In some cases, a **pin.conf** file might be stored in a directory that is not the same as the directory that contains the utility. In that case, you need to run the utility from the directory that contains the **pin.conf** file.
- You can remove or comment out entries in a **pin.conf** file. In that case, the default value is used. If an entry is not in the file, you can add it.

For more information about **pin.conf** files, see *BRM System Administrator's Guide*.

About Editing an Infranet.properties File

Use **Infranet.properties** files to configure BRM Java applications. **Infranet.properties** files are text files, and can be edited by using any text editor.

Infranet.properties entries are name/value pairs, for example:

```
infranet.threadpool.size=3
```

Note the following points when editing **Infranet.properties** files:

- **Infranet.properties** files are stored in the same directory as the executable that uses them.
- If an entry is not in the file, you can add it.

About Editing Business Parameters

Business parameters are configurations that you can use to change BRM functionality to support your business. For example, you can use a business parameter to enable or disable payment incentives.

Business parameters are stored in **/config/business_params** objects in the database. Unlike **pin.conf** files and **Infranet.properties** files, there are no text files that you can open and edit. Instead, you need to use the **pin_bus_params** utility to create a file to edit. To edit a business parameter you:

1. Run the **pin_bus_params** utility to create an XML file.
2. Edit the file.
3. Run the **pin_bus_params** utility to load the XML file.

The XML file includes entries such as this:

```
<EnableCorrectiveInvoices>enabled</EnableCorrectiveInvoices>
```

This example is used to enable or disable the corrective invoicing feature.

To verify that the file was loaded correctly, read the **/config/business_params** object by using the **testnap** utility or Object Browser to verify that all fields are correct.

Loading Configuration Data Into the BRM Database

Some BRM functionality requires that you load configuration into the BRM database. For example, to configure how to calculate the due date of a bill, you edit the **pin_payment_term.xml** file and load it by using the **load_pin_payment_term**.

A typical command looks like this:

```
load_pin_payment_term pin_payment_term.xml
```

Note the following points when editing and loading data:

- When you load a file, it creates a **/config** object in the database. All of the existing data in the **/config** object is replaced. You cannot append new data to the existing data.
- To connect to the BRM database, the utility needs a **pin.conf** file in the directory from which you run the utility. See "About Editing pin.conf Files".
- The file you edit is typically in the same directory as the utility executable file. If it is not in the same directory, include the entire path in the command line when you run the utility.
- In addition to looking in log files, you can verify that data was loaded by reading the **/config** object by using Object Browser or the **testnap** utility.
- The data files are either plain text files or XML files. For XML files, the schema files in the *BRM_home*/**xsd** directory.
- To get help about the utility, use the command with the **-h** parameter.

Verifying That Data is Loaded

To verify that data was loaded into the BRM database, display the object you loaded by using one of the following features:

- Object Browser application
- **robj** command with the testnap utility

For information about reading an object and writing its contents to a file, see *BRM Developer's Guide*.