# Oracle® Communications Billing and Revenue Management

## Collections Manager

Release 15.2
G35834-01
January 2026

ORACLE

Oracle Communications Billing and Revenue Management Collections Manager, Release 15.2

G35834-01

# Contents

# About This Content

This guide describes how to use Oracle Communications Billing and Revenue Management (BRM) Collections Manager to manage collections for your business.

**Audience**

This guide is intended for anyone who installs, configures, or administers Collections Manager and for those who manage collections for your business.

1

# Understanding Collections Manager

Learn how to manage collections for your business by using Collections Manager, in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:



## About Collections

*Collections*, or debt management, is a proactive process used by businesses to collect overdue payments from their customers. Collections include identifying accounts that have overdue balances, determining whether those accounts meet predefined criteria for action, and then taking those actions.

Every business defines its own processes, but collections generally involves a series of increasingly serious steps taken against the customer:

- There is usually a minimum amount due that qualifies for collections. Very small amounts may not be worth the effort required to collect them.
- In most cases, businesses allow a grace period after payment is due. If payment is received within the grace period, no action is taken.
- After the grace period ends, a first action such as a reminder on an invoice, a dunning letter, or a phone call takes place. A late fee may also be assessed.
- If no payment is received as a result of the first collections action, additional steps can be taken. These actions range from additional letters or phone calls to stopping service to referring the case to a collections agency.
- An account is removed from collections when the overdue amount and any late fees are paid or when the debt is written off.

## About Collections Manager

Collections Manager is an optional product that manages in-house debt collection. Collections Manager performs all of its activities at the bill unit (**/billinfo** object) level rather than at the account level. When an account has multiple bill units, Collections Manager targets only those bill units that have overdue balances, but it ignores the account's bill units that do not qualify for collections.

Collections Manager provides tools that enable you to:

- Configure collections criteria
- Determine which bill units have overdue balances
- Apply your collections criteria to those bill units
- Perform automatic collections activities, such as imposing late fees and sending dunning letters
- Manage manual collections activities, such as phone calls by collections agents, or converting payments to installments
- Replace a billing scenario based on the severity

Collections Manager includes these main components:

- **Collections Configuration Center**: A web-based GUI application used to define collections scenarios and actions, set up aging buckets, and define user roles for collections personnel. See *Collections Configuration Center Online Help*.
- **Collections Reports**: A comprehensive reporting feature that you use to display information such as the bill unit status, the delinquent amount, and the total balance for all bill units in collections. Reports can be displayed in summary or detailed form. See "Collections Manager Reports" in *BRM Reports*.
- **Utilities**: Command-line applications that perform specific collections tasks:
  - The **pin_collections_process** utility is a daily batch process that identifies bill units that should be placed in collections and performs automatic collections actions.
  - The **pin_collections_send_dunning** utility is a daily batch process that generates dunning letters based on data collected by **pin_collections_process**. Dunning letters can be sent via email or printed.
  - The **pin_load_template** utility loads dunning letter templates into the database.
  - The **load_config** utility loads additional criteria to be used for assigning a bill unit to a collections scenario.
- **Collections Manager Facilities Modules (FMs)**: Opcodes that are called by the collections applications and utilities. There are standard opcodes that perform the basic collections functionality and policy opcodes that you use to customize that functionality. See "Collections Manager FM Standard Opcodes" and "Collections Manager FM Policy Opcodes" in *BRM Opcode Guide*.
- **Storable classes**: New and extended classes, including **/config** subclasses, that store collections-related data.

Collections agents and collections managers perform day-to-day collections tasks by using Billing Care.

You assign collections roles (agent and manager) in Collections Configuration Center. All collections personnel must have customer service representative (CSR) accounts.

Collections agents monitor and run collections activities for the bill units assigned to them. Using Billing Care, collections agents can:

- View collections tasks for bill units assigned to them
- Monitor the status of both automatic and manual actions defined by the collections scenario
- Insert new actions for a particular bill unit
- Reschedule actions
- Receive credit card payments

# How Collections Manager Processes Bill Units

The main Collections Manager process (triggered by the **pin_collections_process** utility) searches the BRM database to find bill units with overdue balances and bill units that are already in collections.

> **Note**
>
> You should run **pin_collections_process** every day to ensure that bill units enter collections promptly. The timing of actions, such as invoice reminders, dunning letters, and late fees, is based on the day the bill unit enters collections. See "About Overdue and Entry Dates".

Collections Manager processes bill units in two phases:

- During collections preprocessing, the bill unit is evaluated to determine if it should enter, exit, or remain in collections.
- During scenario processing, a bill unit that enters or remains in collections is processed based on its collections profile. These profiles are used only for collections and are not the same as the customer profiles associated with accounts.

## Collections Manager Preprocessing

During preprocessing, Collections Manager determines whether an account's bill unit should enter, exit, or remain in collections. Preprocessing differs depending on whether the bill unit is already in collections.

- **For bill units already in collections**, the process compares the current overdue balance to the exit criteria you defined in the collections scenario (see "About Scenario Processing"). If the overdue balance is less than or equal to the exit criteria for the bill unit's scenario, the bill unit is removed from collections. If the overdue balance is greater than the exit criteria, the bill unit remains in collections and enters scenario processing.

  For example, if the exit criteria is an overdue balance less than or equal to $10, a bill unit with an overdue balance of $8 would be removed from collections, and a bill unit with an overdue balance of $20 would remain in collections.

- **For bill units not in collections**, the process compares the current overdue balance to the minimum collections value that you defined. (See "Setting the Minimum Overdue Balance to Process".)
  - If the overdue balance is less than the minimum collections value, the bill unit remains out of collections and no further action takes place.
  - If the overdue balance is greater than the minimum collections value, the bill unit is assigned to a collections profile based on criteria you defined in advance. By default, all bill units belong to a single collections profile, but you can create any number of custom profiles. For example, you can assign bill units to profiles based on credit score or payment history. See "Setting Up Collections Profiles".

After a bill unit is assigned to a collections profile, the collections process checks if the bill unit meets the entry criteria for any of the scenarios that are valid for the bill unit's profile. The default entry criteria are based on severity, overdue balance and the number of days overdue.

For example, you can specify an overdue balance of $100 that is at least 30 days old. You can define additional parameters for the entry criteria. See "About Collections Scenarios".

- If the bill unit does not meet the entry criteria for any of the scenarios for its collections profile, it remains out of collections and no further action takes place.
- If the bill unit meets the entry criteria for a scenario, it is assigned to that scenario, enters collections, and proceeds to scenario processing.
- If the bill unit meets the entry criteria for more than one scenario, it is assigned to the scenario with the highest entry criteria or the highest severity.

  For example, you might have the following three scenarios for a collections profile:

  – $50 entry requirement and severity 1
  – $100 entry requirement and severity 1
  – $100 entry requirement and severity 2

  A bill unit with an overdue balance of $101 is assigned to the scenario with the $100 requirement with severity 1 because it has the highest entry criteria and the highest severity.
- If you have configured additional parameters for the entry criteria, the bill unit is evaluated for the new scenario. If the new scenario is different from the existing scenario, the bill unit is exited from the existing scenario and is assigned to the new scenario. For example: while in collections, the customer makes a partial payment that reduces the overdue balance sufficiently to allow a scenario replacement to something less severe. The customer service representative (CSR) triggers an automated scenario replacement process that reevaluates the scenario assignment logic on the bill unit and assigns the bill unit to a new scenario.

## About Scenario Processing

A collections scenario defines severity, entry and exit criteria as well as the sequence of actions that take place while an account's bill unit is in collections. See "About Collections Scenarios".

During scenario processing, Collections Manager first determines if any actions are due for this bill unit today. The timing of actions is based on the number of days since the bill unit entered the scenario. (For more information about the dates used in scenario processing, see "About Overdue and Entry Dates".) If an action is due today, Collections Manager then determines whether the action has a **Pending** status. (See "About Collections Action Dependencies".)

- If no actions are due, the process moves to the next bill unit.
- If the scenario includes actions to be carried out today but they have a **Waiting For Dependents** status, the process moves to the next bill unit.
- If the scenario includes actions to be carried out today and they have a **Pending** status, these actions are processed. Some actions happen automatically, some require manual intervention.

  – Automatic actions are performed by Collections Manager. They include predefined actions, such as sending dunning letters or invoice reminders, as well as custom actions that you define. See "Understanding System Actions" and "Understanding Custom Actions".

  – Manual actions, such as making phone calls, or converting payments to installments, are performed by collections agents. Collections agents use Billing Care to track the bill units and actions, either mandatory or optional, for which they are responsible. See "Understanding Manual Actions".

After all scheduled actions have taken place, Collections Manager updates the bill unit's balance and checks to see if the bill unit now meets the exit criteria for the scenario. (It is possible that a payment was received as part of a manual action or that the scenario calls for writing off the balance.)

- If the new overdue balance is less than or equal to the amount specified in the scenario's exit criteria, the bill unit exits the scenario and leaves collections.
- If the overdue balance is still greater than the exit criteria, the bill unit remains in collections.

## About Collections Scenarios

*Collections scenarios* determine the way bill units are handled in Collections Manager. Scenarios define the entry criteria that determine whether a bill unit will be handled by the collections system, the sequence of actions that take place when a bill unit is in collections, the exit criteria that determine when a bill unit leaves collections, and the severity level that determines the priority of the scenario assignment to a bill unit when a bill unit is eligible for two scenarios that have the same entry criteria.

For example, you can define a scenario with the following characteristics:

- A bill unit enters collections with a minimum overdue balance of $100 that is at least 30 days old.
- A bill unit leaves collections when the overdue balance falls below $25.
- The customer receives a courtesy phone call after 10 days in collections.
- A late fee is assessed, and a dunning letter is sent after 30 days in collections.
- A second dunning letter is sent after 45 days.
- The bill unit is closed, and the debt written off after 120 days.
- The bill unit is inactivated after 180 days.
- The severity level that determines the priority of the scenario assignment.

Each scenario includes a list of actions that are performed either:

- On their due date. For example, a manual phone call is made 5 days after the bill unit enters collections, and a dunning letter is sent 10 days after the bill unit enters collections.
- In a specified order. For example, a CSR must make a courtesy phone call to the customer before the system refers the account to a collections agency.

## About Overdue and Entry Dates

There are two important dates that collections managers use when processing bill units:

- The *overdue date* is the due date of the bill that causes the bill unit to enter collections. By default, Collections Manager uses the due date of the latest overdue bill at the time that the bill unit enters collections.

  You can configure Collections Manager to use the earliest due date instead. See "Configuring How Collections Manager Determines Dates".
- The *entry date* is the date on which the bill unit enters a collections scenario. By default, Collections Manager defines the entry date as the overdue date plus the number of days specified in the scenario's entry criteria. For example, if the bill due date is June 15 and the scenario specifies that the bill must be at least 10 days overdue, the entry date is June 25. You can configure Collections Manager to use the actual date on which the bill unit is

processed and enters collections. See "Configuring How Collections Manager Determines Dates".

> **Note**
>
> Do not schedule actions for the same calendar date as the entry date. Scheduling actions on the entry date can cancel the actions.

Collections Manager stores the overdue date in the PIN_FLD_OVERDUE_T field of the **/collections_scenario** object.

Collections Manager uses the overdue date to determine the aging buckets into which a bill unit falls during scenario processing. Aging buckets are used to display information about the age of overdue balances. For example, you could define 30-day, 60-day, and 90-day buckets. The 30-day bucket holds balances that are 1 to 30 days overdue. The 60-day bucket holds balances that are 31 to 60 days overdue, and the 90-day bucket holds balances that are 61 to 90 days overdue. Balances older than 90 days fall into an implicit fourth bucket.

Collections Manager uses the entry date to determine when the actions specified in a scenario take place. For example, you can define a scenario so that a dunning letter is sent 30 days after the bill unit enters the scenario.

> **Note**
>
> If you run Collections Manager irregularly or at long intervals, the overdue and entry dates for a bill unit could be misleading. For example, if you do not run collections for several months, a bill unit that has not paid its bill during this period will enter collections with overdue and entry dates based on the latest of several overdue bills. As a result, collections actions will begin after a lengthy period of nonpayment. To avoid this risk, you should run collections regularly, preferably daily.

## About Collections Action Dependencies

Collections action dependencies ensure that all of a scenario's actions are performed in order and maintain the interval between actions. To do this, Collections Manager:

- Determines whether a scenario's preceding action has been completed or canceled before performing an action on its due date.
- After an action is completed or canceled, reschedules the due date of all of the scenario's remaining actions.

For example, assume a collections scenario includes the following actions:

- Action 1: Collections agent makes a courtesy phone call two business days after the bill unit enters collections.
- Action 2: Collections Manager sends an invoice reminder four business days after the bill unit enters collections.
- Action 3: Collections Manager applies a late fee six business days after the bill unit enters collections.

On day four, Collections Manager determines whether Action 1 has been completed or canceled and, since Action 1 is still open, does not send an invoice reminder. After the collections agent makes the courtesy phone call on day five, Action 1 is marked as completed

and then Collections Manager reschedules Action 2 for seven business days after the bill unit enters collections and Action 3 for nine business days after the bill unit enters collections. This ensures that there are two business days between the courtesy phone call and the invoice reminder, and four business days between the courtesy phone call and the late fee.

Collections Manager creates dependencies between actions in a scenario by using action status settings:

- **Pending**: This status specifies that the action can be run on its due date. All preceding actions have been canceled or completed.
- **Waiting For Dependents**: This status specifies that the action cannot be performed until the preceding action is canceled or completed. If an action is set to this status on its due date, Collections Manager does not perform the action and then moves to the next bill unit.

When a bill unit first enters a collections scenario, Collections Manager sets the first action's status to **Pending** and all other actions' status to **Waiting For Dependents**. After the first action is completed or canceled, Collections Manager updates the second action's status from **Waiting For Dependents** to **Pending** and then reschedules the due date of all remaining actions. After the second action is completed or canceled, Collections Manager updates the third action's status from **Waiting For Dependents** to **Pending** and reschedules the due date of all remaining actions. This process continues for all of the scenario's remaining actions.

## About Collections Groups

Some countries have legal requirements that prevent actions, such as collections, from being taken against minors. This law applies to minors even when they have their own accounts and pay their own bills. Collections groups enable you to define the legal entity, such as a parent or guardian, who is responsible for a bill if it enters into collections.

When you create an account and its bill units, you can specify whether collections actions must be taken against another account and bill unit. To do this, you create a collections group, which consists of the following:

- A parent bill unit: During collections processing, Collections Manager enters only the parent bill unit into a collections scenario. The parent bill unit is assigned a collections scenario based on:
  - The combined overdue balance from all bill units in the collections group. For example, if one child bill unit is $100 past due and a second child bill unit is $50 past due, the overdue balance is $150.
  - The oldest due date from all bill units in the collections group. For example, if one child bill unit had a due date of June 1 and the second child bill unit had a due date of June 10, the overdue date is June 1.
- One or more child bill units: During collections processing, Collections Manager skips all child bill units in the group. The child bill unit and parent bill unit do not need to be linked hierarchically.

A bill unit can belong to only one collections group, and a parent bill unit cannot be a child bill unit of the same collections group. You can change a collections group's parent and add or remove child bill units at any time.

When you define a collections action in Collections Configuration Center, you can specify whether the action applies to the parent bill unit, to all the child bill units, or to the parent and all the child bill units. For example, you can specify that all "Referring the Account to a Collections Agency" actions apply to the parent bill unit only, and all "Sending a Dunning Letter" actions apply to the parent and all child bill units.

## Corrective Billing and Collections

When BRM generates a corrective bill for a bill that is not yet in collections, the Collections Manager uses the sum of the adjustments for the bill to determine whether an account's bill unit should enter collections.

If BRM generates a corrective bill for a bill already in collections, Collections Manager takes the following actions based on the sum of the adjustments for the bill:

- If the sum of the adjustments for the bill goes below the threshold, BRM moves that bill out of collections.
- If the sum of the adjustments for the bill remains above the threshold, BRM maintains the account's bill unit in collections but replaces the bill number of that bill with the bill number of the corrective bill.

# About Defining Rule-Based Collections Scenario Assignment

By default, Collections Manager assigns a collections scenario to each bill unit based on the overdue balance and the number of overdue days. You can use the "**load_config**" utility to load additional parameters for the collections scenarios assignment. This utility loads the parameters into the **/config/collections/scenario_params** object, which is used to define an evaluation formula. Each formula is, in turn, linked to the corresponding collections scenario assigned to the bill unit.

Using Collections Configuration Center, you can add, modify, or delete the values of these additional parameters within the scenario definitions.

# About Collections Scenario Replacement

When a bill unit meets the criteria for a collections scenario, it is assigned to that scenario and enters the collections process. If the bill unit changes, such as when a customer makes a partial payment, it may no longer meet the criteria for the assigned collections scenario. In this case, Collections Manager reevaluates the criteria based on additional parameters and may replace the current collections scenario with a new one.

# Customizing Collections

You can use opcodes to customize collections, for example:

- You can customize dunning letter data before it is stored in the BRM database. For example, you can enrich the standard data with additional information, such as the date when the account will be inactivated.
- You can apply finance charges to overdue amounts.
- You can customize how BRM calculates the collections due date (for example, to account for holidays).
- You can customize how BRM applies late fees to overdue charges.

See "Customizing Collections Manager" in *BRM Opcode Guide*.

2

# Installing and Configuring Collections Manager

Learn how to install the Oracle Communications Billing and Revenue Management (BRM) Collections Manager software. Installation includes both client and server components.

> **Note**
>
> Collections Manager Reports are installed separately. See "Installing BRM Reports" in *BRM Reports*.

Topics in this document:



## Installing and Configuring Collections Manager Server

The Collections Manager server components include opcodes, utilities, and storable classes.

### Installing Collections Manager

You can install Collections Manager in the *BRM_home* directory. For instructions, see "Installing Individual BRM Components" in *BRM Installation Guide*.

### Configuring Collections Manager

This section describes how to configure the Collections Manager software.

#### Updating Partitions

If your event tables are partitioned during installation, run the **partition_utils** utility with the **-o update** parameter from the *BRM_home***/apps/partition_utils** directory:

```
perl partition_utils.pl -o update
```

For more information, see "Updating Partitions" and "partition_utils" in *BRM System Administrator's Guide*.

#### Configuring the pin_bill_day Script to Run Collections

To run collections automatically, you need to edit the **pin_bill_day** script.

1. Open the **pin_bill_day** script in a text editor. The script is located in *BRM_home***\bin**.
2. Find the Collections Manager section of the script. If the following lines are commented out, remove the comment marks:

```
cd ${PINDIR}/apps/pin_collections
pin_collections_process
pin_collections_send_dunning
```

3. Save and close the file.

## Uninstalling BRM Collections Manager

To uninstall BRM Collections Manager, See "Uninstalling Optional Components" in *BRM Installation Guide*.

# Installing Collections Configuration Center

Before installing BRM Collections Configuration Center, you must install:

- BRM Collections Manager. See "Installing and Configuring Collections Manager Server".
- An application such as WinZip for extracting compressed files.

For instructions on how to install Collections Configuration Center, see *Collections Configuration Center Installation Guide*.

3

# Assigning Collections Personnel Roles

Learn how to set up roles in Oracle Communications Billing and Revenue Management (BRM) Collections Manager to control which employees can perform collections actions and view collections details for your customers.

Topics in this document:



## Assigning Collections Personnel Roles

You assign collections roles in Collections Configuration Center. The tasks that collections personnel can perform and the information they can see depend on their role.

Collections Configuration Center displays a list of all CSR accounts. You select the accounts that should have a collections role.

To customize how agents are assigned, use the following:

- PCM_OP_COLLECTIONS_POL_ASSIGN_AGENT opcode. See "Assigning Bill Units Automatically" in *BRM Opcode Guide*.
- PCM_OP_COLLECTIONS_POL_ASSIGN_DCA policy opcode. See "Assigning Bill Units to a Debt Collections Agency" in *BRM Opcode Guide*.

4

# Setting Up Collections Manager

Learn how to set up Oracle Communications Billing and Revenue Management (BRM) Collections Manager.

Topics in this document:



## Adding Custom Pay Types

When you run the **pin_collections_process** utility to trigger collections actions, the utility considers only bill units with the default pay types. If you created a custom payment type, you need to configure collections to support it.

To add custom pay types:

1. Open the **pin_collections** configuration file (*BRM_home***/apps/pin_collections/pin.conf**) in a text editor, where *BRM_home* is the directory in which the BRM server software is installed.
2. Enter the custom BRM pay type code in the **pay_type** entry:

   ```
   - pin_collections_process pay_type PaymentType
   ```

   For example, enter **10013** for the wire transfer payment type:

   ```
   - pin_collections_process pay_type 10013
   ```

   To add multiple custom pay types, separate the *PaymentType* entries by a comma.

   ```
   - pin_collections_process pay_type 10013,10014
   ```

   > **Note**
   >
   > You can find valid pay type codes in the *BRM_home***/include/pin_cust.h** file, under PAY_TYPE.

3. Save and close the file.

# Setting the Minimum Overdue Balance to Process

You can specify the minimum overdue balance that triggers Collections Manager. When you run **pin_collections_process**, only bill units with overdue balances greater than or equal to the amount you specify are considered.

> **Note**
>
> The value you specify in the collections **pin.conf** file is a system-wide minimum value. Collections scenarios cannot have entry criteria lower than this value. For example, if the minimum value in the collections **pin.conf** file is $20 and a collections scenario has an entry criteria of $10, a bill unit will enter collections only if its overdue balance is $20 or more.

To set the minimum overdue balance for collections:

1. Open the **pin_collections** configuration file (*BRM_home***/apps/pin_collections/pin.conf**) in a text editor.
2. Set the **minimum_due** entry to the minimum overdue balance:

   ```
   - pin_collections_process minimum_due Amount
   ```

   For example, enter **25** to specify $25:

   ```
   - pin_collections_process minimum_due 25.00
   ```

   > **Note**
   >
   > The default is **0.0**.

3. Save and close the file.

# Setting the Number of Bill Units Retrieved during Step Searches

The Collections Facilities Module (FM) opcodes use step search when searching for bill units. The step size determines the number of bill units retrieved at one time. You can improve performance by changing this value.

To specify the number of bill units to retrieve in step searches:

1. Open the Connection Manager (CM) configuration file (*BRM_home***/sys/cm/pin.conf**) in a text editor.
2. Set the **account_search_batch** entry to the number of bill units.

   ```
   - fm_collections account_search_batch Amount
   ```

   For example, enter **500** to retrieve 500 bill units with each step of the search.

   ```
   - fm_collections account_search_batch 500
   ```

> **Note**
>
> The default is **1000**.

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

# Setting Up Invoice Reminders

Collections Manager uses the Universal Message Store (UMS) framework to deliver invoice reminders. See "Using BRM Messaging Services" in *BRM Developer's Guide* for general information.

To use invoice reminders as part of the collections process, you must complete three tasks related to the UMS framework:

- Enable messaging by modifying the PCM_OP_INV_POL_PREP_INVOICE policy opcode. See "Enabling Messaging" in *BRM Developer's Guide*.
- Load a message style sheet (**message.xsl**) that makes it possible to display messages in invoices. See "Loading the Message Style Sheet" in *BRM Developer's Guide*.
- Load invoice reminder templates into **/string** objects in the BRM database. See "Creating and Loading Message Templates" in *BRM Developer's Guide*. Collections Manager includes a sample invoice reminder template that you can modify and rename as necessary. The file is *BRM_home***/sys/msgs/messages/message_invoice_reminder.en_US**.

# Creating Dependencies between Collections Actions in a Scenario

By default, Collections Manager performs automated collections actions on their due date, regardless of whether the preceding collections actions have been completed or canceled. This can cause collections actions to be completed out of order.

You can configure Collections Manager to create dependencies between collections actions in a scenario and thus perform collections actions in order. In this case, Collections Manager performs the following additional steps during scenario processing:

- Performs a collections action on its due date only if the action's status is set to **Pending**.
- After a collections action is completed or canceled, automatically reschedules the due dates of all outstanding collections actions in the scenario. To do so, Collections Manager calculates the difference in days between the action's completed or canceled date and the action's due date, and then moves the due dates of all outstanding actions by the calculated number of days. For example, assume a collections scenario contains two collections actions:
  - Action 1: A manual phone call from an agent 5 days after the bill unit enters collections.
  - Action 2: Referring the case to an outside collections agency 10 days after the bill unit enters collections.

  If action 1 does not occur until day 7, Collections Manager changes the due date of action 2 from day 10 to day 12. This maintains the interval between Action 1 and Action 2.

To create dependencies between collections actions in a collections scenario:

1. Open the CM configuration file (*BRM_home*/**sys/cm/pin.conf**) in a text editor.
2. Set the **collections_action_dependency** entry to **1**:

   ```
   - fm_collections collections_action_dependency 1
   ```

   > **Note**
   >
   > The default is **0**, which disables dependencies between collections actions.

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

# Adding Custom Collections Reason Codes for Billing Care

In Billing Care, you can perform operations such as inserting new collections actions, rescheduling collections actions, and exempting bill units from collections.

For many of these operations, you choose a reason that is associated with the action. Billing Care includes a default set of reasons, which you can modify.

To modify the default reasons, you can add or change reason codes in each **reasons.locale** file. See "Loading Localized or Customized Strings" in *BRM Developer's Guide* for more information.

5

# Setting Up Dunning Letters

Learn how to set up Oracle Communications Billing and Revenue Management (BRM) to send collections dunning letters to your customers.

Topics in this document:



## About Setting Up Dunning Letters

To set up Collections Manager to generate dunning letters, you must run Email Data Manager (DM) and load your dunning letter templates.

You can customize dunning letter data before it is stored in the BRM database. For example, you can enrich the standard data with additional information, such as the date on which the account will be inactivated. To do so, customize the PCM_OP_COLLECTIONS_POL_PREP_DUNNING_DATA opcode. See *BRM Opcode Guide*.

PCM_OP_COLLECTIONS_POL_PREP_DUNNING_DATA enables customization of dunning letter data before it is stored in the database. You can modify the type of data gathered by customizing the opcode. For example, you can enrich the standard data with additional information, such as the date on which the account will be inactivated.

## Loading Dunning Letter Templates

You load dunning letter templates by using the **pin_load_template** utility. These dunning letter templates are used in Collections Configuration Center to set the dunning letters action and the invoice reminders action.

Dunning letter templates are XSL documents that provide the basic content of the letter, leaving placeholders for specific information such as the customer's name, overdue balance, and so on. You can design and load multiple dunning letter templates for different scenarios and situations.

Collections Manager includes two sample dunning letter templates: **dunning_first.xsl** and **dunning_last.xsl**. Both templates are stored in *BRM_home***/sys/data/config/stylesheets**. You can use a third-party XSL editor to modify these templates or to create your own templates.

After creating your template, load it into the BRM database by using the **pin_load_template** utility. Templates are available in Collections Configuration Center immediately after you load them. See "Defining Collections Actions".

To load a dunning letter template:

1. Go to a directory that contains a valid configuration (**pin.conf**) file.

   > **Note**
   >
   > A configuration file is installed automatically in *BRM_home***/sys/data/config**.

   The **pin_load_template** utility uses information in the configuration file to connect to the BRM database. See "Connecting BRM Utilities" in *BRM System Administrator's Guide*.

2. Load the XSL template by entering the following command:

   ```
   pin_load_template -brand brand_POID - name template_name -type dunning -
   locale locale_name -template file_name -usexsl
   ```

   where:

   - *brand_POID* is the Portal object ID (POID) of the root account.
   - *template_name* is the name of the template as it should appear in Collections Configuration Center.
   - *locale_name* is the BRM locale of the template. See "Locale Names" in *BRM Developer's Guide*.
   - *file_name* is the name and full path of the template file.

For example, the following command loads the **dunning_first.xsl** template for the root account:

```
pin_load_template -brand "0.0.0.1/account 1" -type dunning -locale en_US -template
BRM_home/config/dunning_first.xsl -name DunningTemplate -usexsl
```

# Setting Dunning Letter Delivery Preferences for Non-invoice Bill Units

You specify how to deliver dunning letters to *non-invoice* bill units by using the **pin_collections** configuration file (*BRM_home***/apps/pin_collections/pin.conf**).

For non-invoice bill units, you can set the following delivery options:

- Whether to deliver the dunning letter as hardcopy or by email.
- For email delivery, whether to send the dunning letter as part of the message body or as an attachment.
- For dunning letters sent as email attachments, the file path to use for providing customized content in the email body.

# Setting the Delivery Option

By default, dunning letters are sent by email. You can also use the **pin_bus_params** utility to print out the dunning letters and send them as hard copy. For information about this utility, see "pin_bus_params" in *BRM Developer's Guide*.

To set the delivery option for dunning letters:

1. Go to *BRM_home***/sys/data/config**.

2. Use the following command to create an editable XML file from the collections instance of the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsCollections bus_params_collections.xml
   ```

   This command creates an XML file named **bus_params_collections.xml.out** in your current directory. If you do not want this file in your current directory, specify the path as part of the file name.

3. In **bus_params_collections.xml.out**, set **DeliveryPreference** to **hard_copy**:

   ```
   <DeliveryPreference>hard_copy</DeliveryPreference>
   ```

   The default value is **email**.

   > ⚠ **Caution**
   >
   > BRM uses the XML in this file to overwrite the existing instance of the **/config/business_params** object. Use care when updating parameters in the file.

4. Save and exit the file.
5. Rename the **bus_params_collections.xml.out** file to **bus_params_collections.xml**.
6. Use the following command to load your changes into the **/config/business_params** object:

   ```
   pin_bus_params bus_params_collections.xml
   ```

   You should run this command from the *BRM_home*/**sys/data/config** directory, which includes support files used by the utility. To run it from a different directory, see "pin_bus_params" in *BRM Developer's Guide.*

7. Read the object with the **testnap** utility or the Object Browser to verify that all fields are correct.

   For general instructions on using **testnap**, see "Using the testnap Utility to Test BRM" in *BRM Developer's Guide*. For information on how to use Object Browser, see "Reading Objects" in *BRM Developer's Guide*.

8. Stop and restart the Connection Manager (CM).

   For more information, see "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.

# Setting the Email Delivery Preference

By default, dunning letters are delivered in the body of the email.

To set the email delivery preference for dunning letters:

1. Open the *BRM_home*/**apps/pin_collections/pin.conf** file in a text editor.
2. Change the value of the **email_option** entry:
   - To deliver dunning letters as email attachments, enter **1**.
   - To deliver dunning letters in the email body, enter **0**. This is the default.

     For example:

     ```
     - pin_collections_send_dunning email_option 0
     ```

3. Save and close the file.

# Specifying a File for the Email Body

When you send dunning letters as email attachments, you can include customized information in the email body. You specify the path to a text file for the content of the message.

To specify the path to the message content:

1. Open the *BRM_home*/**apps/pin_collections/pin.conf** file in a text editor.
2. Set the path in the **email_body** entry:

   ```
   - pin_collections_send_dunning email_body Path
   ```

   For example, the following entry specifies the path to the **letter** file:

   ```
   - pin_collections_send_dunning email_body ./letter
   ```

   > **Note**
   >
   > This option takes effect only if **email_option** is set to **1**.

3. Save and close the file.

6

# Defining Collections Actions

Learn about collections actions, in Oracle Communications Billing and Revenue Management (BRM), which are individual steps that are performed in the process of collecting overdue balances from your customers, such as applying a late fee or making a phone call.

Topics in this document:



## About Defining Collections Actions

*Collections actions* are individual steps performed in the process of collecting overdue balances. An action might be applying a late fee or making a phone call.

You can define as many actions as you need. For example, you can define different phone call actions for various situations, such as courtesy calls, warning calls, and so on. The same action can be included in multiple scenarios.

When you define a collections action, you specify the following in Collections Configuration Center:

- The collections action name
- The action type: manual, one of several system types, or custom
- For bill units in a collections group, the target bill units: the specified bill unit only (Self), the parent and all child bill units (All Members and Parent), or all child bill units (All Members)

> **Note**
>
> The action target for each customer action remains consistent across all scenarios. Once the action target mode is established for an action, it cannot be altered for any newly defined scenarios.

## Understanding Manual Actions

Manual actions are completed by collections agents. When a scenario calls for a manual action, BRM saves information about the action in the database. Billing Care displays this information so that a collections agent can complete the action.

For example, suppose that a scenario calls for the customer to receive a courtesy phone call when the account's bill unit is 10 days into the collections process. When a bill unit enters collections, this phone call is included in the list of actions for the bill unit. On the day when the phone call is due, the phone call appears in the list of tasks for the collections agent assigned to the bill unit.

The most common manual action is a phone call, but you can create any others that your business requires.

# Understanding System Actions

System actions are performed automatically by BRM when either of the following two utilities is run:

- **pin_collections_process**
- **pin_deferred_act** (as a part of the **pin_bill_day** script)

If an error occurs during any collections action performed when **pin_collections_process** is run, the status of the collections action object (**/collections_action**) and the schedule object (**/schedule**) corresponding to the collections action is updated to **Error**.

If an error occurs during any collections action performed when **pin_deferred_act** is run, the status of the collections action remains **Pending** but the schedule object corresponding to the collections action is updated to **Error**. When **pin_collections_process** is run and the collections actions are processed again, if an error occurs, then the collections action object is updated to **Error**.

The status of collections actions is updated to **Completed** on successful completion of the actions.

By default, Collections Manager supports the following types of system collections actions:

- **Charging a late fee**. For late fees, you specify the currency type, such as US dollars, and the late fee type, either a specified amount or a percentage of the overdue amount.
- **Adding a finance charge**. For finance charges, you specify to charge a certain percentage of the overdue amount.
- **Sending a dunning letter**. For dunning letters, you specify the dunning letter template to use. You can have different templates for different purposes. For example, you could define several dunning letter templates with varying levels of severity. These templates could be used at different times in the same scenario or in separate scenarios.
- **Sending an invoice reminder**. For invoice reminders, you specify the invoice reminder template to use. You can have different templates for different purposes. For example, you could define several invoice reminder templates with varying levels of severity. These templates could be used at different times in the same scenario or in separate scenarios.

You can customize each system action's behavior by modifying policy opcodes. See the chapter on collections opcode workflows in *BRM Opcode Guide* for information.

# Performing Scheduled Collections Actions After Reinstalling Collections Manager

By default, if you run the **pin_collections_process** utility after reinstalling Collections Manager, the scheduled collections actions that have a due date prior to the date on which you reinstalled Collections Manager are not performed.

To perform all scheduled collections actions irrespective of whether they have a due date prior to the date or after the date on which you reinstalled Collections Manager and ran **pin_collections_process**:

1. Open the CM configuration file (*BRM_home***/sys/cm/pin.conf**) in a text editor.
2. Add the following entry:

   ```
   - fm_collections execute_all_actions value
   ```

   where *value* is:

   - **1** to perform all collections actions irrespective of whether they have a due date prior to the date or after the date on which you run **pin_collections_process** after reinstalling Collections Manager.
   - **0** to perform only those collections actions that have a due date later than the date on which **pin_collections_process** is run after reinstalling Collections Manager. This is the default.
3. Save and close the file.
4. Stop and restart the CM.

# Understanding Custom Actions

Custom actions are system actions that you create. You define custom actions and descriptions in Collections Configuration Center and implement the actions by modifying the PCM_OP_COLLECTIONS_POL_EXEC_POLICY_ACTION policy opcode. See "Performing Custom Collections Actions" in *BRM Opcode Guid*e.

# Increasing the Size of the CM Cache for Actions Data

When defining actions, you might need to increase the space allocated to the action data in the CM cache.

To increase the CM cache size for actions data:

1. Open the CM configuration file (*BRM_home***/sys/cm/pin.conf)**.
2. Increase *cache_size* in the following entry:

   ```
   - cm_cache fm_collections_config_actions_cache number_of_entries,
   cache_size, hash_size
   ```

   > **Note**
   >
   > The default is 40960 bytes.

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

7

# Defining Collections Scenarios

Learn about collections scenarios, in Oracle Communications Billing and Revenue Management (BRM), which are an ordered list of actions that are taken against a bill unit that is in collections.

Topics in this document:



## Defining Collections Scenarios

When you create a new scenario, you decide which collections profile it applies to. You can define multiple scenarios with different characteristics for the same profile. For example, you could set up one scenario for relatively small overdue balances and another for larger balances.

A single scenario cannot be used with multiple collections profiles. If you want to implement scenarios with the same functionality across several profiles, you can define multiple scenarios with the same characteristics but different names.

A scenario includes an ordered list of actions that are taken against the bill unit. When you define a scenario, you select from a list of actions that you have previously defined and then specify the following characteristics:

- The order in which the actions should take place.
- The number of days after entering collections that the action should be completed. For example, 5 days after entering collections.
- Whether an action is optional or mandatory.

> **Note**
>
> If you define a collections scenario that includes both a write-off and a bill unit inactivation, the write-off must come before the inactivation. When a bill unit is inactivated, an event with a credit balance is created to cancel the remaining service for the bill unit. The cycle fees are then prorated, and the remaining credit is returned to the bill unit, in some cases as a separate item. If the bill unit has already been inactivated, the write-off will fail because of the unallocated item. In this case, a customer service representative (CSR) must manually allocate the item so that it can be written off.

## Loading Additional Parameters for Scenario Assignment

You load additional parameters for a collections scenario assignment by using the **load_config** utility. See "load_config" in *BRM Developer's Guide* for more information.

The utility takes an XML file as input and creates or updates the **/config/collections/scenario_params** object in the BRM database. Billing Care reads the **/config/collections/**

**scenario_params** object and displays the parameters while defining the evaluation formula to associate with a collections scenario.

To load additional collections scenario parameters:

1. Open the *BRM_home*/**sys/data/config/config_collections_scenario_params.xml** file in a text editor.
2. Edit the file, which includes examples and instructions. Table 7-1 describes the parameters in this file.

**Table 7-1 Elements Used to Store Additional Scenario Parameters**

| Element | Description |
|---|---|
| **PARAM_NAME** | Name of the parameter. |
| **STRING_ID** | The string ID of the parameter name. This field is used for localization. |
| **STR_VERSION** | The version of the localized string within the domain. This field is used for localization. |
| **TYPE** | The type of value that the parameter can be assigned, from one of the following types:<br>• **1**: STR (alphanumeric)<br>• **2**: INT (integer)<br>• **3**: ENUM (indicating that the parameter is one of an ordered list of possible values. An array of values must be provided for this selection.)<br>• **4**: DECIMAL<br>• **5**: TSTAMP (timestamp)<br>For example, to provide a set of possible values, you set the **<TYPE>** element to **3**, and enter an array of values for this parameter in the **<VALUES>** element |
| **CLASS_NAME** | The storable class object associated with the parameter. |
| **FIELD_NAME** | The field inside the storable class referenced in the **<CLASS_NAME>** element. To enter the field name of a field present under an array or a substruct, use the format <FIELD_NAME>*array*.*field*</FIELD_NAME >, which includes the parent field of the attribute separated by periods (.). For example: <FIELD_NAME>PIN_FLD_COLLECTIONS_PARAMS.PIN_FLD_CREDIT_PROFILE</FIELD_NAME >. |
| **VALUES** | An array list of values that the parameter can assume. The **<VALUES>** array list is present only if the selection for the **<TYPE>** element is **3**. |

For example, the following entry defines a new parameter called **Payment Method**:

```
<PARAMS elem="0">
     <PARAM_NAME>Payment Method</PARAM_NAME>
     <TYPE>3</TYPE>
     <CLASS_NAME>/billinfo</CLASS_NAME>
     <FIELD_NAME>pay_type</FIELD_NAME>
     <VALUES elem="0">
               <NAME>CC</NAME>
               <VALUE>10003</VALUE>
     </VALUES>
     <VALUES elem="1">
               <NAME>DD</NAME>
               <VALUE>10005</VALUE>
```

```
     </VALUES>
     <VALUES elem="2">
               <NAME>CASH</NAME>
               <VALUE>10011</VALUE>
     </VALUES>
     <VALUES elem="3">
               <NAME>CHECK</NAME>
               <VALUE>10012</VALUE>
     </VALUES>
</PARAMS>
```

In this example:

- The name of the parameter is **Payment Method**.
- The type of value is **3** (which is ENUM, and so an array of values follows).
- The storable class is **/billinfo**.
- The field inside the storable class object is **pay_type**.
- The **Values** array lists the four possible payment methods: **CC**, **DD**, **Cash**, and **Check**.

For example, the following entry defines a parameter called **Credit Profile**:

```
<PARAMS elem="3">
     <PARAM_NAME>Credit Profile</PARAM_NAME>
     <TYPE>2</TYPE>
     <CLASS_NAME>/profile/collections_params</CLASS_NAME>
     <FIELD_NAME>PIN_FLD_COLLECTIONS_PARAMS.PIN_FLD_CREDIT_PROFILE</FIELD_NAME>
</PARAMS>
```

In this example:

- The name of the parameter is **Credit Profile**.
- The type of value is **2** (which is INT).
- The storable class is **/profile/collections_params**.
- The field including the path to the storable class object is PIN_FLD_COLLECTIONS_PARAMS.PIN_FLD_CREDIT_PROFILE.

By default, the **config_collections_scenario_params.xml** file contains sample data for the following parameters:

- Payment Method (Credit Card, Direct Debit, cash, check) from the **/billinfo** storable class
- Account Status from the **/account** storable class
- Account Type from the **/account** storable class

To define additional data during scenario evaluation, you can create and maintain the required data (parameters and their values) in a customized storable class or by extending the **/profile** storable class. This storable class should have a link to the bill unit to which the data is applicable. There can be only one record per bill unit. The parameter used should be at the 0th level of the storable class.

3. Save and close the file.
4. Open the *BRM_home*/**apps/load_config/pin.conf** file in a text editor.
5. Uncomment the following entry:

   ```
   - load_config validation_module libLoadValidCollections  LoadValidCollections_init
   ```

6. Save and close the file.

7. Load the updated file by running the following command:

   ```
   load_config -v config_collections_scenario_params.xml
   ```

   > **Note**
   >
   > - The **load_config** utility requires a configuration (**pin.conf**) file.
   > - If you do not run the utility from the directory in which the configuration file is located, include the complete path to the file. For example,
   >
   >   ```
   >   load_config -v BRM_home/sys/data/config/
   >   config_collections_scenario_params.xml
   >   ```

8. Stop and restart the CM.

   To verify that the updated collections parameter configurations were loaded, you can display the **/config/collections/scenario_params** object by using Object Browser, or use the **robj** command with the **testnap** utility.

# Increasing the Size of the CM Cache for Scenario Data

When evaluating scenarios, you might need to increase the space allocated to the scenario data in the CM cache.

To increase the CM cache size for scenario data:

1. Open the CM configuration file (*BRM_home***/sys/cm/pin.conf)**.
2. Increase *cache_size* in the following entry:

   ```
   - cm_cache fm_collections_config_scenario_cache number_of_entries,
   cache_size, hash_size
   ```

   > **Note**
   >
   > The default is 40960 bytes.

   If the file does not have this entry, add it.
3. Save and close the file.
4. Stop and restart the CM.

8

# Setting Up Collections Profiles

Learn how to create collections profiles to organize bill units for collections purposes in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:



## Setting Up Collections Profiles

*Collections profiles* categorize bill units for collections purposes. By default, profiles classify bill units based on currency. However, you can customize them to categorize bill units according to other attributes, such as credit score. For example, bill units with scores below a certain threshold can be assigned to one collections profile, while those with higher scores can be assigned to a different profile.

A bill unit's collections profile helps determine which collections scenario it is assigned to. For example, bill units in the low-credit-score profile may be assigned to scenarios with more aggressive collections actions and lower entry criteria. Conversely, bill units with high credit scores might be placed in scenarios with more lenient policies.

By default, all bill units belong to a single collections profile. If you do not need a more complex system, you do not need to define any additional profiles and can map all scenarios to the default profile.

To set up additional profiles, complete the following tasks:

1. Define collections profiles in Collections Configuration Center by specifying the profile name, description, and currency. See "Managing Collections Profiles" in *Collections Configuration Center Online Help*.

   When you create a new profile, Collections Configuration Center creates a **/config/collections/profile** object, which stores profile definitions.

2. Customize the PCM_OP_COLLECTIONS_POL_SELECT_PROFILE policy opcode to define the characteristics of collections profiles and map bill units to them. See "Mapping Bill Units to Collections Profiles" in *BRM Opcode Guide*.

> **Note**
>
> This step requires programming.

## Increasing the Size of the CM Cache for Profiles Data

When setting up additional profiles, you might need to increase the space allocated to the profiles data in the CM cache.

To increase the CM cache size for profiles data:

1. Open the CM configuration file (*BRM_home***/sys/cm/pin.conf**) in a text editor.
2. Increase *cache_size* in the following entry:

   ```
   - cm_cache fm_collections_config_profiles_cache number_of_entries,
   cache_size, hash_size
   ```

   > **Note**
   >
   > The default is 20480 bytes.

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

9

# Managing Promise-to-Pay Agreements

Learn how to manage promise-to-pay agreements in Oracle Communications Billing and Revenue Management (BRM) by using Collections Manager. Your customers can enter into promise-to-pay agreements, giving them more time to pay off their debt while deferring collection actions.

Topics in this document:



> **Note**
>
> If a customer wants to pay a bill in multiple installments when it is not in collections, use installment payments instead. See "Setting Up Payment Installments" in *BRM Configuring and Collecting Payments*.

## About Promise-to-Pay Agreements

When a bill unit is in collections, the customer can request a promise-to-pay agreement. This agreement defers all collection actions for a bill unit, allowing the customer to pay the due amount at a date beyond the original payment due date.

Customers can pay the agreed-upon amount in a single payment or in multiple installments. You can set up the installment schedule and amount in the following ways:

- Automatically by Collections Manager. See "About System-Configured Promise-to-Pay Installments".
- Manually, According to the Customer. See "About Manually Configured Promise-to-Pay Installments".

> **Note**
>
> You cannot change a promise-to-pay agreement after it is created.

When a customer enters into a promise-to-pay agreement, Collections Manager reschedules all outstanding collection actions in the bill unit to start one day after the last promise-to-pay

installment due date. For example, if the last installment due date is 20 March, the outstanding collection actions are scheduled to start 21 March and onwards.

If the customer pays off the total promise-to-pay amount on time, the outstanding collection actions are canceled, and the bill unit is removed from collections. If the customer cancels or breaches the promise-to-pay agreement, the outstanding collection actions are resumed the day after the agreement is canceled or breached.

## About System-Configured Promise-to-Pay Installments

You use system-configured promise-to-pay installments to have Collections Manager calculate the installment schedule and installment amount for you. Collections Manager divides the total amount due into equal installments at equal intervals, such as four installments of $50 due every 30 days.

To create system-configured promise-to-pay installments, you provide the following information:

- The total amount the customer agrees to pay. The total amount cannot be less than the minimum amount to exit collections.
- The due date for the customer's first promise-to-pay installment.
- Either the amount due for each installment, or the total number of installments.

  If the installment amount is provided, Collections Manager calculates the number of installments by dividing the total amount due by the installment amount. For example, if a customer owes $200 and requests a $50 installment amount, the total number of installments is 4.

  If the number of installments is provided, Collections Manager calculates the installment amount as the total amount due divided by the number of installments. For example, if a customer owes $300 and requests 3 installments, the installment amount is $100.
- Either the interval between each installment, or the number of days until the total amount due will be paid off.

  If the interval is provided, Collections Manager calculates the agreement's total number of days by multiplying the interval by the number of installments. For example, if the interval is 15 days and the number of installments is 4, the agreement's total number of days is 60.

  If the agreement's total number of days is provided, Collections Manager calculates the interval as the total number of days divided by the number of installments. For example, if the total number of days is 30 and the number of installments is 2, the interval is 15 days.

> **Note**
>
> If the calculated interval includes a partial day, such as 12.5 days, the extra days are added to the last installment interval: 12 days, 12 days, 12 days, 14 days.

For example, a customer enters into a promise-to-pay agreement to pay off $500 by making $100 installment payments every 14 days, starting 1 June. In this case, Collections Manager creates the promise-to-pay installments shown in Table 9-1.

**Table 9-1 Sample System-Configured Promise-to-Pay Installments**

| Installment Number | Amount Due | Interval from Previous Installment | Due Date |
|---|---|---|---|
| 1 | $100 | N/A | 1 June |
| 2 | $100 | 14 Days | 15 June |
| 3 | $100 | 14 Days | 29 June |
| 4 | $100 | 14 Days | 13 July |
| 5 | $100 | 14 Days | 27 July |

# About Manually Configured Promise-to-Pay Installments

> **Note**
>
> BRM allows you to create manually configured promise-to-pay installments only if your system contains promise-to-pay specifications (**/config/collections/promise_to_pay_spec** objects).

Manually configured promise-to-pay installments allow you to create installments with varying amounts due and varying intervals. For example, a customer could request three installments: the first for $200 due on 1 January, the second for $100 due 15 days later, and the third for $150 due 20 days later.

You specify the limits for manually configured installments by creating a promise-to-pay specification, which defines the following:

- The minimum initial installment amount is a percentage of the total amount due, such as 25%.
- The minimum amount for a promise-to-pay installment such as 10% of the total amount due or $50.
- The maximum number of installments allowed, such as 5.
- The maximum number of days between promise-to-pay installments, such as 30 days.
- Whether to temporarily increase the customer's credit limit for the promise-to-pay agreement and then decrease the credit limit after each installment payment.

  For example, assume a customer has a $200 credit limit and enters into a $100 promise-to-pay agreement on 1 March, with a $30 installment due 10 March and a $70 installment due 30 March. If you specify to increase credit limits temporarily, the customer's credit limit would increase on 1 March to $300 ($200 + $100), decrease after the 10 March payment to $270 ($300 – $30), and then decrease after the 30 March payment to $200 ($270 – $70).

For information about creating promise-to-pay specifications, see "Creating Promise-to-Pay Specifications".

To create manually configured promise-to-pay installments, you provide the following information:

- The total amount the customer agrees to pay. The total amount cannot be less than the minimum amount to exit collections.

- The due date for the customer's first promise-to-pay installment.
- The promise-to-pay specification to apply for validation.
- For each installment, the absolute amount due or the percentage of the total amount due. The sum of all installments must equal 100% of the total amount due.
- For each installment, the interval, in days, from the previous installment.

For example, a customer could enter into a promise-to-pay agreement to pay off $500 starting 1 June, with the installments shown in Table 9-2.

**Table 9-2 Sample Manually Configured Promise-to-Pay Installments**

| Installment Number | Amount Due | Interval from Previous Installment | Due Date |
|---|---|---|---|
| 1 | $250 | N/A | 1 June |
| 2 | $50 | 30 Days | 30 June |
| 3 | $100 | 14 Days | 14 July |
| 4 | $50 | 7 Days | 21 July |

# About the Promise-to-Pay Status

When your customers enter into a promise-to-pay agreement, Collections Manager begins tracking the agreement and each installment. Table 9-3 describes the available statuses for an overall promise-to-pay agreement.

**Table 9-3 Overall Status of a Promise-to-Pay Agreement**

| Status | Description |
|---|---|
| **Pending** | The customer has entered into a promise-to-pay agreement, but the due date for the first installment has not yet occurred. |
| **Canceled** | The customer has canceled the promise-to-pay agreement. All outstanding collection actions are rescheduled to start from the next day. |
| **Completed** | The customer has paid off the full amount owed in the promise-to-pay agreement. |
| **Kept** | All installments to date have been paid by the customer. Other installments are due in the future. |
| **Broken** | The customer missed an installment, breaking the promise-to-pay agreement. All outstanding collection actions are rescheduled to start from the next day. |

Collections Manager stores the overall status of promise-to-pay agreements in the PIN_FLD_PROMISE_TO_PAY field of **/collections_scenario** objects.

Table 9-4 describes the available statuses for an individual promise-to-pay installment.

**Table 9-4 Status of a Promise-to-Pay Installment**

| Status | Description |
|---|---|
| **Pending** | The installment is due in the future. |

**Table 9-4 (Cont.) Status of a Promise-to-Pay Installment**

| Status | Description |
|---|---|
| **Canceled** | The installment has been canceled. |
| **Completed** | The installment was paid off in full on time. |
| **Error** | Collections Manager was unable to retrieve the installment's status. |
| **Broken** | The customer did not pay off the installment by the due date. |

You can view the status of a promise-to-pay installment by using Billing Care. See "Viewing a Bill Unit in Collections" in *Billing Care Online Help*.

# About Paying Promise-to-Pay Agreements Early

If your customers pay off their promise-to-pay agreement early, Collections Manager changes the status of all promise-to-pay installments to **Completed**. Likewise, the status of the overall promise-to-pay agreement is changed to **Completed**.

Collections Manager also cancels all remaining collection actions and removes the customer's bill from collections.

# About a Missed Promise-to-Pay Installment

When a customer does not pay off a promise-to-pay installment by its due date, Collections Manager does the following:

- Changes the status of the missed promise-to-pay installment to **Broken**.
- Changes the status of all remaining promise-to-pay installments to **Canceled**.
- Changes the status of the overall promise-to-pay agreement to **Broken**.
- Reschedules all remaining collection actions to start from the next day.

For example, Table 9-5 shows how the status and schedule of collection actions change when a customer misses a 15 May promise-to-pay installment.

**Table 9-5 Sample Collections Action Status and Schedule for a Missed Promise-to-Pay Installment**

| Collections Action | Collections Due Date | Status | Rescheduled Due Date |
|---|---|---|---|
| Promise-to-Pay Installment | 15 April | Completed | N/A |
| Promise-to-Pay Installment | 15 May | Broken | N/A |
| Promise-to-Pay Installment | 15 June | Canceled | N/A |
| Apply Late Fee | 16 June | Pending | 16 May |
| Apply Finance Charge | 20 June | Pending | 20 May |
| Inactivate Services | 25 June | Pending | 25 May |

# About Partial Promise-to-Pay Payments

When customers pay only a partial amount of a promise-to-pay installment, the installment remains in **Pending** status, and the customer still owes the remaining balance by the due date. If the installment is not paid in full by the due date, its status is changed to **Broken** and treated as a missed installment payment.

# About Canceling a Promise-to-Pay Agreement

Your customers can cancel an existing promise-to-pay agreement with your company. Customers might do this, for example, if they realize that they will be unable to pay their installments on time. When an agreement is canceled, Collections Manager changes the status of all pending promise-to-pay installments to **Canceled**. It also reschedules all remaining collections actions to start from the next day.

# Setting Up Collections Manager for Promise-to-Pay Agreements

To set up Collections Manager for promise-to-pay agreements, do the following:

1. Create promise-to-pay specifications for manually configured agreements. See "Creating Promise-to-Pay Specifications".
2. (Optional) Perform customizations to the promise-to-pay process by using the Collections policy opcodes. See "Customizing the Promise-to-Pay Process".
3. (Optional) Configure BRM to generate reminder messages for customers a specified amount of time before their promise-to-pay installment is due. You do so by:
   - Setting up BRM to send notifications to customers through an external notification application. See "Sending Messages to Customers through External Notification Applications" in *BRM Managing Customers*.
   - Running this utility command on a regular basis: **pin_gen_notifications -collections_action promise_to_pay**. See "pin_gen_notifications" in *BRM Managing Customers*.

After Collections Manager is set up, you can start creating promise-to-pay agreements for your customers in collections. See "Creating Promise-to-Pay Agreements".

## Creating Promise-to-Pay Specifications

Promise-to-pay specifications define the limits for manually configured promise-to-pay installments. For example, they specify the minimum installment amount, the maximum interval between installments, and so on.

You create promise-to-pay specifications by:

1. Editing the *BRM_home***/sys/data/config/config_collections_promise_to_pay_spec.xml** file.
2. Loading the XML file into the BRM database by using the **load_config** utility.

   See "load_config" in *BRM Developer's Guide* for information about the utility's syntax and parameters.

Table 9-6 describes the elements in the **config_collections_promise_to_pay_spec.xml** file for defining a promise-to-pay specification.

**Table 9-6 Promise-to-Pay Specification Elements**

| Element | Description |
|---|---|
| **<PROMISE_TO_PAY_SPEC>** | An array that defines one promise-to-pay specification. Add an array for each specification that you want to create. |
| **<MINIMUM>** | The minimum installment amount. |
| **<TYPE>** | The type of the minimum promise-to-pay amount: absolute amount (**0**) or percentage of the total amount due (**1**). The default is **0**. |
| **<MIN_PERCENTAGE_THRESHOLD>** | The minimum amount due for the initial installment as a percentage of the total amount due. |
| **<MAX_UNEQUAL_INSTALLMENTS>** | The maximum number of installments that are allowed. |
| **<MAX_UNEQUAL_INTERVALS>** | The maximum interval, in days, between each installment. |
| **<CREDIT_LIMIT_FLAG>** | Whether to temporarily increase a customer's credit limit during the promise-to-pay period (**1**) or to leave the customer's credit limit unchanged (**0**). The default is **0**. |

The following shows sample content for the **config_collections_promise_to_pay_spec.xml** file that creates one promise-to-pay specification with these limits:

- Each installment must be at least $25.
- The initial promise-to-pay installment must be at least 20% of the total amount due. For example, if the total amount due is $200, the initial installment amount must be $40 or higher.
- The maximum number of installments is 10, and the maximum number of days between installments is 50.
- The customer's credit limit will not be adjusted during the promise-to-pay period.

```
<ObjectList
   xmlns="http://www.oracle.com/schemas/BusinessConfig"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.oracle.com/schemas/
BusinessConfig ../../../xsd/config_collections_promise_to_pay_spec.xsd">
   <ConfigObject>
      <DESCR>Collections promise-to-pay specification</DESCR>
      <NAME>Sample promise-to-pay specification</NAME>
      <PROGRAM_NAME>load_config</PROGRAM_NAME>
      <PROMISE_TO_PAY_SPEC elem="0">
         <MINIMUM>25</MINIMUM>
         <TYPE>0</TYPE>
         <MIN_PERCENTAGE_THRESHOLD>10</
MIN_PERCENTAGE_THRESHOLD>
         <MAX_UNEQUAL_INSTALLMENTS>10</MAX_UNEQUAL_INSTALLMENTS>
         <MAX_UNEQUAL_INTERVALS>50</MAX_UNEQUAL_INTERVALS>
         <CREDIT_LIMIT_FLAG>0</CREDIT_LIMIT_FLAG>
      </PROMISE_TO_PAY_SPEC>
   </ConfigObject>
</ObjectList>
```

To load the **config_collections_promise_to_pay_spec.xml** file into the BRM database:

1. Open the *BRM_home*/**apps/load_config/pin.conf** file in a text editor.
2. Uncomment the following entry by removing the #:

   ```
   #- load_config validation_module libLoadValidCollections  LoadValidCollections_init
   ```

3. Save and close the file.
4. Load the XML file by running the following command:

   ```
   $PIN_HOME/apps/load_config/load_config -v $PIN_HOME/sys/data/config/
   config_collections_promise_to_pay_spec.xml
   ```

5. Stop and restart the CM.

   To verify that the specifications were loaded, you can display the **/config/collections/promise_to_pay_spec** object by using Object Browser or by using the **robj** command with the **testnap** utility.

## Customizing the Promise-to-Pay Process

You can customize the promise-to-pay process in the following ways by using Collections policy opcodes:

- How Collections Manager handles missed promise-to-pay installments by using the PCM_OP_COLLECTIONS_POL_HANDLE_BREACH_PROMISE_TO_PAY policy opcode.
- The conditions a payment must meet to set the status of a promise-to-pay installment to **Completed** by using the PCM_OP_PYMT_POL_EXEC_COLLECTIONS_ACTION policy opcode.

For more information, see "Managing Promise-to-Pay Agreements" in *BRM Opcode Guide*.

## Creating Promise-to-Pay Agreements

You can create a promise-to-pay agreement for a customer in collections by using the following:

- Billing Care. See "Working with Promise-to-Pay Agreements" in *Billing Care Online Help*.
- A custom client application calling the promise-to-pay collections opcodes. See "Managing Promise-to-Pay Agreements" in *BRM Opcode Guide*.
- A custom client application calling the **Create a Promise-to-Pay Agreement for a Bill Unit** operation in the Billing Care REST API. For more information, see *REST API Reference for Billing Care*.

10

# Configuring How Collections Manager Determines Dates

Learn how to set the overdue and entry dates that Oracle Communications Billing and Revenue Management (BRM) Collections Manager uses for bills that are in collections.

Topics in this document:



## Configuring How Collections Manager Determines Dates

You can configure how Collections Manager determines the overdue and entry dates it uses (described in Table 10-1). For information about how Collections Manager uses these dates, see "About Overdue and Entry Dates".

**Table 10-1 Overdue and Entry Dates**

| Option | Description |
|---|---|
| Overdue date | The overdue date is one of the following:<br>• (Default) The overdue date is the due date of the latest overdue bill when the bill unit enters collections. This date is not updated as long as the bill remains in collections.<br>• The overdue date is the due date of the oldest overdue bill when the bill unit enters collections, even if the overdue balance from that bill is not sufficient to meet the scenario's entry criteria.<br>See "Configuring the Overdue Date" for instructions. |
| Entry date | The entry date is one of the following:<br>• (Default) The entry date is the overdue date plus the number of days specified in the scenario's entry criteria. This date is updated if the overdue date changes.<br>• The entry date is the actual date on which a bill unit is processed and enters collections. This date is not updated as long as the bill unit remains in collections. If Collections Manager is run at irregular or long intervals with this setting, the entry date can differ significantly from the overdue date.<br>See "Configuring the Entry Date" for instructions. |

To customize how due dates are set, use the PCM_OP_COLLECTIONS_POL_CALC_DUE_DATE opcode. See "Customizing Due Dates" in *BRM Opcode Guide*.

The following examples illustrate the overdue and entry dates for four possible combinations of configuration options. The examples are based on the following assumptions:

- There is a monthly charge of $15, due on the 15th of the month.

- The collections scenario specifies a minimum overdue balance of $20, with a minimum of 10 days late.
- No payments are received in January, February, or March, but a $15 payment is received before the due date in April.
- Collections Manager runs every day.
- The tables show the overdue and entry dates as of the end of each month.

**Case 1: Default Settings for Both Dates**

Table 10-2 shows the results of the default setting for both dates. The overdue and entry dates stay the same as long as the bill unit remains in collections.

**Table 10-2 Default Date Example**

| Option | Jan | Feb | Mar | Apr |
|---|---|---|---|---|
| Payment received | None | None | None | 15.00 |
| Overdue amount | 15.00 | 30.00 | 45.00 | 45.00 |
| Overdue date | None | Feb 15 | Feb 15 | Feb 15 |
| Entry date | None | Feb 25 | Feb. 25 | Feb 25 |

**Case 2: Optional Settings for Both Dates**

Table 10-3 shows the results of the optional setting for both dates. When the bill unit enters collections, the overdue date is set to January 15, even though the overdue balance for that month was not sufficient to trigger the scenario. Also, note that the overdue date changes when a payment eliminates the January overdue balance.

**Table 10-3 Optional Date Example**

| Option | Jan. | Feb. | Mar. | Apr. |
|---|---|---|---|---|
| Payment received | None | None | None | 15.00 |
| Overdue amount | 15.00 | 30.00 | 45.00 | 45.00 |
| Overdue date | None | Jan. 15 | Jan. 15 | Feb. 15 |
| Entry date | None | Feb. 25 | Feb. 25 | Feb. 25 |

**Case 3: Default Setting for Overdue Date and Optional Setting for Entry Date**

In Table 10-4, the dates match those for Case 1, but the entry date is determined by the actual processing date. If the collections process was not run daily, the entry date would not necessarily match the date specified by the scenario's entry criteria.

**Table 10-4 Processing Date Example**

| Option | Jan. | Feb. | Mar. | Apr. |
|---|---|---|---|---|
| Payment received | None | None | None | 15.00 |
| Overdue amount | 15.00 | 30.00 | 45.00 | 45.00 |
| Overdue date | None | Feb. 15 | Feb. 15 | Feb. 15 |
| Entry date | None | Feb. 25 | Feb. 25 | Feb. 25 |

#### Case 4: Optional Setting for Overdue Date and Default Setting for Entry Date

In Table 10-5, the entry date changes when the overdue date is updated to reflect the payment. This would delay any remaining collection actions defined in the scenario.

**Table 10-5 Varied Date Settings**

| Option | Jan. | Feb. | Mar. | Apr. |
|---|---|---|---|---|
| Payment received | None | None | None | 15.00 |
| Overdue amount | 15.00 | 30.00 | 45.00 | 45.00 |
| Overdue date | None | Jan. 15 | Jan. 15 | Feb. 15 |
| Entry date | None | Jan. 25 | Jan. 25 | Feb. 25 |

# Configuring the Overdue Date

To configure how Collections Manager determines the overdue date:

1. Open the CM configuration file (*BRM_home*/**sys/cm/pin.conf**) in a text editor.
2. Configure the **old_overdue_behavior** entry:

   ```
   - fm_collections old_overdue_behavior value
   ```

   where *value* is:

   - **0**: Specifies to use the latest overdue bill date. This is the default.
   - **1**: Specifies to use the earliest overdue bill date.
3. Save and close the file.
4. Stop and restart the CM.

# Configuring the Entry Date

To determine how Collections Manager determines the entry date:

1. Open the **pin_collections** configuration file (*BRM_home*/**apps/pin_collections/pin.conf**) in a text editor.
2. Configure the following entry:

   ```
   - pin_collections_process use_current_time value
   ```

   where *value* is:

   - **0**: Specifies to use the entry date configured in the collections scenario. This is the default.
   - **1**: Specifies to use the current date.
3. Save and close the file.

11

# Integrating Collections Manager with Custom Client Applications

Learn how to integrate your custom application with Oracle Communications Billing and Revenue Management (BRM) Collections Manager.

Topics in this document:



## About Integrating Collections Manager with Custom Client Applications

To enable a custom client application to manage collection activities:

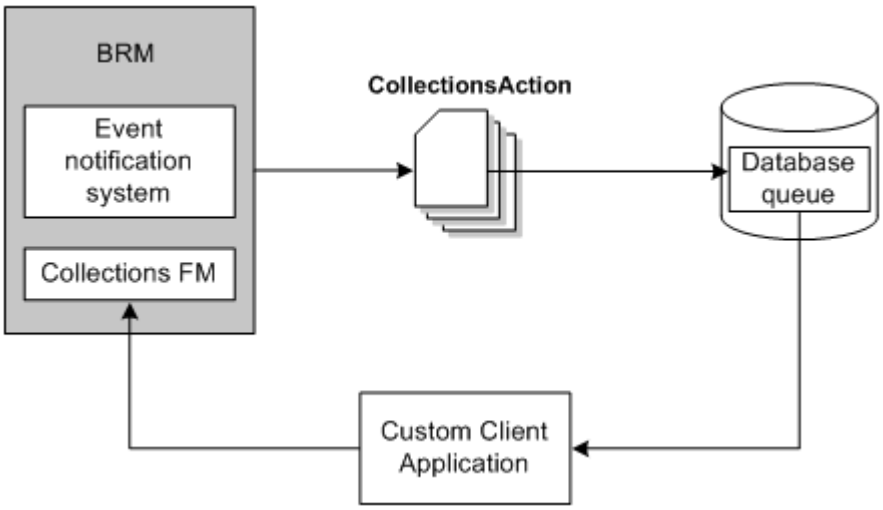
- Configure BRM to send collection notification events to your custom client application. See "Sending Collections Notifications to Custom Client Applications".
- Customize your client application to call the Collections opcodes. See "Collections Opcode Workflows" in *BRM Opcode Guide*.
- (Application Integration Architecture systems only) Configure the **pin_collections_process** utility to publish a **CollectionsInfoChange** business event.

  See "Creating CollectionsInfoChange Business Events".

You can integrate Collections Manager with a custom client application, such as Siebel CRM, so customer service representatives (CSRs) can track and manage collections activities directly in the custom client application.

The custom client application manages collection activity, such as preparing dunning letters or applying late fees, by calling the Collections FM. Collections Manager notifies the custom client application when a collections activity occurs, such as an account entering collections, by using the BRM event notification system. Figure 11-1 shows how data is passed.

**Figure 11-1 How Custom Client Applications Communicate with Collections Manager**




To integrate Collections Manager with a custom client application:

- Configure the custom client application to call the Collections Manager opcode. See "About Calling the Collections Manager API".
- Configure Collections Manager to notify the custom client application when collection information changes. See "About Notifying Custom Client Applications when Collections Activity Occurs".

# About Calling the Collections Manager API

CSRs need to perform the following collection tasks in their custom client application:

- Manage the collection of overdue balances
- Retrieve information about collection activities
- Perform manual collection activities
- Prepare dunning letters for customers

To perform these tasks, configure your client application to accept the required information and pass it to the Collections Manager opcodes. You can determine the information required by looking at the target opcode's input flist. For more information, see "Collections Opcode Workflows" in *BRM Opcode Guide*.

> **Note**
>
> You can pass information to Collections Manager opcodes either directly from the custom client application or through Oracle Application Integration Architecture (Oracle AIA). For more information about Oracle AIA, see "Using BRM with Oracle Application Integration Architecture" in *BRM Developer's Guide*.

# About Notifying Custom Client Applications when Collections Activity Occurs

Collections Manager uses event notifications to alert custom client applications when collections activities occur, such as when an account enters collections or a dunning letter is sent. Collections Manager sends an alert in the form of an **/event/audit/collections/action** notification event, which includes information about the collections action, such as:

- The POID of the **/collections_action** object
- The action type: manual, custom, or system
- A description of the action
- The action status

To notify custom client applications when collection activity occurs:

- Configure BRM to publish **/event/audit/collections/action** notification events. See "Sending Collections Notifications to Custom Client Applications".
- Customize your client application to retrieve notification events from the database queue.

# Creating CollectionsInfoChange Business Events

If your custom client application connects to BRM through Oracle Application Integration Architecture (Oracle AIA), BRM must publish **CollectionsInfoChange** business event to the AQ database queue after a collections job has run.

To create a **CollectionInfoChange** business event:

1. Open the **pin_collections_process** configuration file (*BRM_home***/apps/pin_collections/pin.conf**) in a text editor.
2. Uncomment the following entry and ensure it is set to **1**:

   ```
   - pin_collections_process publish_run_details 1
   ```

3. Save and close the file.

# Sending Collections Notifications to Custom Client Applications

You can configure BRM to synchronize collections data with your custom client application. To do so, you must:

- Create Oracle Advanced Queuing (AQ) database queues
- Configure event notification to trigger calls to opcodes when **/event/audit/collections/action** and **/event/notification/collections/info_change** events occur
- Configure the Payload Generator EM to build **CollectionsAction** and **CollectionsInfoChange** business events and send them to the Oracle DM
- Map the **CollectionsAction** and **CollectionsInfoChange** business events to your database queues

To set up your system to send collections notifications to custom client applications, follow the instructions in "Configuring Your AQ Database Queues" in *BRM System Administrator's Guide* using the configuration files shown in Table 11-1.

**Table 11-1 Configuration Files for Collections Notifications**

| File Type | File Name and Location | Action |
|---|---|---|
| Event notification file | *BRM_home*/**sys/data/config/pin_notify_collections** | Use this file or merge it with your existing event notification file, and then load the final file into the database. |
| Payload configuration file | *BRM_home*/**sys/eai_js/payloadconfig_collections.xml** | Use this file or merge it with your existing payload configuration file. |
| Business event-to-queue mapping file | *BRM_home*/**sys/dm_oracle/aq_event_map** | Uncomment these business events in the file:<br>• **CollectionsAction**<br>• **CollectionsInfoChange** |

12

# Collections Manager Utilities

Learn about the syntax and parameters for the Oracle Communications Billing and Revenue Management (BRM) Collections Manager utilities.

Topics in this document:



## pin_collections_process

Use this utility to find bill units (**/billinfo** objects) that are in collections and trigger collection actions defined by the bill unit's collections scenario.

You must set a minimum value that this utility uses to determine whether bill units are considered for collections. See "Setting the Minimum Overdue Balance to Process".

By default, this utility is commented out in the **pin_bill_day** script. You should run **pin_collections_process** daily before "pin_collections_send_dunning". See "Running Billing Scripts" in *BRM Configuring and Running Billing* for more information about the billing scripts.

> **Note**
>
> To connect to the BRM database, **pin_collections_process** requires a configuration (**pin.conf**) file in the directory from which you run the utility. See "Connecting BRM Utilities" in *BRM System Administrator's Guide*.

### Location

*BRM_home***/bin**

### Syntax

```
pin_collections_process  [-billinfo BillInfoPOID] [-record_failure true|false] [-
pay_type PayType]
                         [-debug] [-verbose] [-report] [-help]
```

### Parameters

**-billinfo *BillInfoPOID***
Triggers collection actions against the specified bill unit. Use this parameter to perform collections against a single bill unit that belongs to a collections group.

**-record_failure true|false**
When set to **true**, the utility records details of failed operations and transactions that occurred while it runs in **/request/failed** objects.

**-pay_type *PayType***
Triggers collection actions for the specified payment types, such as direct debit and credit card. Replace *PayType* with the payment type's element ID, such as **10001** (invoice), **10002** (checking account debit), and **10003** (credit card). Delimit multiple payment types with a comma.

**-debug**
Sets the log level to debug and outputs debug information into the log file for this process. If not set, only error-level information is output.

**-verbose**
Displays progress information.

**-report**
Shows the summary of this billing run, including how many account bill units are in collections, how many exited collections, and which system actions were performed.

**-help**
Displays the syntax and parameters for this utility.

**Results**

The collection status of all bill units with overdue balances is evaluated and, if necessary, updated. For bill units that enter or remain in collections, all actions defined by the relevant collections scenario are performed.

# pin_collections_send_dunning

Use this utility to email or print dunning letters. The "pin_collections_process" utility calls **pin_collections_send_dunning**, but you can also run it independently.

You specify the invoice delivery method, email or postal, for bill units in Billing Care or Customer Center:

- Email: **pin_collections_send_dunning** emails the dunning letter.
- Postal: **pin_collections_send_dunning** prints the dunning letter.

For other payment types, you specify the delivery option in the collections configuration (**pin.conf**) file. See "Setting Dunning Letter Delivery Preferences for Non-invoice Bill Units".

> **Note**
>
> The Email Data Manager must be running when you run this utility, even for printing. For information, see "Sending Email to Customers Automatically" in *BRM Managing Customers* and "Configuring the Email Data Manager for Printing" in *BRM Designing and Generating Invoices*.

The data used to create the dunning letter is generated by "pin_collections_process" and stored in the database. This data is combined with a dunning letter template to make the final letter.

> **Note**
>
> To connect to the BRM database, **pin_collections_send_dunning** requires a configuration (**pin.conf**) file in the directory from which you run the utility.

### Location

*BRM_home***/bin**

### Syntax

```
pin_collections_send_dunning [-record_failure true|false] [-export Path] [-verbose]
```

### Parameters

**-record_failure true|false**
When set to **true**, the utility records details of failed operations and transactions that occurred while it runs in **/request/failed** objects.

**-export *Path***
Exports the generated dunning letters to the specified directory.

**-verbose**
Displays progress information.

### Results

The dunning letters whose data is collected by "pin_collections_process" are prepared and sent using the delivery method for each bill unit.

# pin_load_template

Use this utility to load dunning letter templates into **/config/invoice_templates/dunning** objects. These objects contain the XSL templates that you use to format dunning letters and invoice reminders.

For additional information, see "Loading Dunning Letter Templates".

> **Note**
>
> To connect to the BRM database, **pin_load_template** requires a configuration (**pin.conf**) file in the directory from which you run the utility. A configuration file is installed automatically in *BRM_home***/sys/data/config** when you install BRM Collections Manager.

### Location

*BRM_home***/bin**

### Syntax

```
pin_load_template [-brand poid] -name template_name [-type template_type] -format
template_output_format
```

```
                -locale locale_string -template template_file [-usexsl] [-debug] [-
logfile log_file]
```

### Parameters

**-brand *poid***
Specifies the root account POID for which you are loading the template, such as **0.0.0.1 / account 1**.

**-name *template_name***
Specifies the template name displayed in Collections Configuration Center. The name must be unique in a given brand.

> **Note**
>
> If the name contains spaces, use quotation marks.

**-type *template_type***
Specifies the template category, such as **dunning** or **invoice**. The default is **invoice**.

**-format *template_output_format***
Specifies the output format for the dunning letter templates.

**-locale *locale_string***
Specifies the locale string. The default is **en_US**.

**-template *template_file***
Specifies the template file name and path.

**-usexsl**
Specifies the use of XSL to format dunning letters. Must be used when loading an XSL dunning letter template.

**-debug**
Logs the flist and detailed messages in the log file.

**-logfile *log_file***
Specifies the full path and name of the log file.

### Results

Loads the specified file into the BRM database as a **/config/invoice_templates/dunning** object.

The utility fails if the specified file or path is invalid.