Oracle® Communications Billing and Revenue Management

Managing Accounts Receivable

Release 15.2

G35854-01

January 2026

Oracle Communications Billing and Revenue Management Managing Accounts Receivable, Release 15.2

G35854-01

Contents

About This Content

This guide describes how to use and manage accounts receivable (A/R) data in Oracle Communications Billing and Revenue Management (BRM).

Audience

This guide is intended for operations personnel and system administrators.

1 Making Adjustments

Learn how to adjust your customers' balances after they have been charged incorrectly in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

To adjust multiple accounts in bulk, see "Adjusting Multiple Accounts Simultaneously".

About Adjustments

An adjustment is a transaction that debits or credits a customer's account by changing the amount due for a bill item, or the amount of a noncurrency balance.

- A credit adjustment decreases the customer's balance; that is, it decreases the amount a customer owes. A credit adjustment is represented as a negative number. For example, when you give 100 free minutes, the adjustment is represented as -100.
- A debit adjustment increases the customer's balance.

Customer service representatives perform adjustments on a variety of levels. For example:

- Event level: If the customer made a 10-minute call that was mistakenly billed as a 30-minute call, CSRs perform the adjustment for that specific call at the event level.
- Account level: If the customer's package provided 100 included minutes a month, but charges started accruing after only 30 minutes, CSRs perform the adjustment at the account level instead.

Credit adjustments do the following:

- For currency balances, decrease the Due of a bill item
- For noncurrency balances, increase the adjusted balance

If a CSR is crediting an event, the balance impact of that event is removed from the customer's account. Debit adjustments have the opposite effect.

The way that BRM processes adjustments and records the adjustment's balance impact varies from level to level, as follows:

- **A/R and individual account**: Adjustments at this level reduce the current balance of the customer's bill. The account's default balance group is decreased by the amount of the credit.
- **Subscription service and member service**: Adjustments at these levels are similar to account adjustments. However, the adjustment targets a specific balance group associated with the subscription service or member service rather than using only the default balance group. In this case, BRM uses the balance group supplied by the **/service** object associated with the subscription service or member service selected by the CSR.

As with account adjustments, the CSR must allocate the adjustment before it affects the customer's bill.

- **Bill level**: Adjustments at this level reduce the current balance of the customer's bill. Here, the amount of the adjustment is subtracted from the due amount for the bill, and payment for that amount is not requested.

  Adjustments can be made to an entire bill or a selection of bill items, distributing the adjustment as a fixed amount per item or as a percentage. In either case, BRM creates a single adjustment item and transfers the credit to the bill items covered by the adjustment. The current balance of the appropriate balance groups is reduced by the amount of the credit.

  Bill adjustments act against A/R bills only. CSRs cannot adjust a bill from a nonpaying bill unit (**/billinfo** object) by filing a bill adjustment directly against that bill. Instead, they file the bill adjustment against the parent A/R bill. Also, the adjustment amount cannot exceed the total amount of the bill against which the adjustment is applied.
- **Item level**: When you adjust a bill item, the amount of the adjustment is subtracted from the Due of the bill item, and payment for that amount is not requested. The current balance of the appropriate balance group is reduced by the amount of the credit.
- **Event level**: Adjustments at this level depend on whether the adjustment occurs before billing or after billing. In either case, the original event is never adjusted.
  - If the adjustment occurs before billing has run, it changes the balance impact of the shadow event.
  - If the adjustment occurs after billing has run, it changes the balance impact of the adjustment event (**/event/billing/adjustment/event**).

Table 1-1 summarizes how adjustments handle currency and noncurrency balances.

Table 1-1 Adjustable Balances

Adjustment Type	Currency (2)	Noncurrency
Account adjustment (1)	Yes	Yes
Subscription service adjustment (1)	Yes	Yes
Member service adjustment (1)	Yes	Yes
Bill adjustment	Yes	No
Item adjustment	Yes	No
Event adjustment	Yes	Yes

Note:

1. For these adjustments, there must be a noncurrency balance group at the account level for the adjustment to affect a noncurrency balance.
2. You cannot adjust loan sub-balances in a currency balance.

An event adjustment that credits currency reduces the balance impact of the event and its General Ledger (G/L) impact. It does not cancel the event, only the cost of the event.

Note

- When adjusting pending items, ensure proper G/L reporting by specifying that BRM create a shadow adjustment instead of a standard adjustment. See "Adjusting Events" in *BRM Opcode Guide*.
- BRM does not support adjustment reversals. It also does not allow performing a new adjustment on an existing adjustment event or item.

Enabling Accurate Handling of Event-Level Adjustments

You can configure BRM to ensure that any adjustment amount exceeding the corresponding item's due is left unallocated, while a zero-amount transfer event is created if the item due is zero. To do so, you set the **EventAdjustmentUnallocation** business parameter.

- When the business parameter is disabled, BRM allows adjustments if the item's due amount has reached zero due to non-event-level actions, which can result in a negative due amount.
- When the business parameter is enabled, BRM allows adjustments if the item's due amount has reached zero due to non-event-level actions but leaves the amount unallocated.

To enable accurate handling of event-level adjustments:

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

   ```
   pin_bus_params -r BusParamsAR bus_params_AR.xml
   ```

3. Open the **bus_params_ar.xml.out** file.
4. In the XML file, add the below entry:

   ```
   <EventAdjustmentUnallocation>enabled</EventAdjustmentUnallocation>
   ```

5. Save the file as **bus_params_AR.xml**.
6. Load the XML file into the BRM database:

   ```
   pin_bus_params -v bus_params_AR.xml
   ```

7. Stop and restart the CM.

Applying Event Rounding Rules to Adjustments

You can configure BRM to apply your event rounding rules to event adjustments. For example, if you configured usage events with six-decimal precision, any event-level adjustments on usage fees are also rounded to six-decimal precision.

To apply event rounding rules to event adjustments:

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

2. Create an XML file from the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsAR bus_params_AR.xml
   ```

3. Open the **bus_params_ar.xml.out** file.
4. In the XML file, add the below entry:

   ```
   <UseEventRoundingRulesForAdjustment>enabled</
   UseEventRoundingRulesForAdjustment>
   ```

5. Save the file as **bus_params_AR.xml.**
6. Load the XML file into the BRM database:

   ```
   pin_bus_params -v bus_params_AR.xml
   ```

7. Stop and restart the CM.

2

Adjusting Multiple Accounts Simultaneously

Learn how to adjust multiple paying accounts simultaneously in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

Adjusting Multiple Accounts Simultaneously

In some situations, you may want to adjust multiple paying accounts simultaneously. For example, if regular daily rates were mistakenly applied during a holiday discount period, you would most likely want to perform adjustments for all accounts that used the service over the holiday.

This type of adjustment is called a bulk adjustment. To perform a bulk adjustment, create a file that lists each account that should receive the bulk adjustment and provides information about the adjustment. You then run the **pin_apply_bulk_adjustment** utility to perform the adjustments based on the data in the file.

The file also specifies whether the adjustment is taxable. BRM bases tax calculation on the full adjustment amount for each account. Depending on how adjustment taxation is configured, BRM may apply the adjustment tax reversal when the bulk adjustment runs or send the adjustment amount for deferred taxation during the next billing run.

When BRM processes a bulk adjustment, it performs each adjustment as a separate transaction. Therefore, there is no need to roll back the successful adjustments if some adjustments fail. BRM reports any accounts that failed the bulk adjustment in a log file. All CSV records that failed to process due to an incorrect format or due to server-side errors are written to an error file. The default name of this file is **pin_mta_failed.csv**. The names and locations of these files can be defined in the **pin_apply_bulk_adjustment pin.conf** file.

The file is in CSV format and is typically supplied by an external source program. CSV is a standard file format that uses commas as field delimiters and line breaks as record delimiters. CSV files should contain no blank lines. CSV files used for bulk adjustments are typically generated by an external program, and the records must use the following format:

```
account_POID, adjustment_amount, balance_group_POID, tax_flag, tax_code,
tax_supplier_ID, balance_element_ID, end_time, reason_code_domain, reason_code,
description
```

> **Note**
>
> A CSV record should not contain the line break implied above. Line breaks should occur at the end of each record.

Consider these guidelines:

- The *account_POID*, *adjustment_amount*, and *balance_element_ID* fields are mandatory. You can omit any of the other fields. If you omit any field in a record (including those at the end of the record), you must still include the associated comma so BRM can keep track of which field it is processing.
- If you omit the *balance_group_POID*, BRM adjusts the default balance group.
- The *tax_flag* field indicates whether the adjustment requires tax reversal. If the flag is set to **1**, the adjustment does not need tax reversal. If the flag is set to **2**, the adjustment includes a tax reversal. If you omit this field, the adjustment occurs without a tax reversal.
- The *balance_element_ID* is the numeric code for the balance element you are adjusting (for example, **840** for US dollars). You can specify currency or noncurrency balance elements.
- The format for the *end_time* field is *MM/DD/YYYY*. BRM uses midnight for the date you specify as the time stamp for the adjustment.
- If you include a *reason_code_domain*, you must also include a *reason_code*, and the reverse.
- If the description field includes any commas, you must enclose the field in quotation marks ("). Quotation marks are optional for description fields that do not include commas.

The following CSV file segment shows a bulk adjustment with tax:

```
0.0.0.1 /account 15269 0, -9.5, 0.0.0.1 /balance_group 12901 0, 2, , , 840, , , , Rate
issue
0.0.0.1 /account 12581 0, -9.5, 0.0.0.1 /balance_group 16165 0, 1, , , 840, , , , Rate
issue
0.0.0.1 /account 15557 0, 10, , , , , 1000010, 04/26/2004, 12, 5, "Service drop, fix
this"
```

In the preceding example, the first record specifies an adjustment with a tax reversal against the currency balance element for a specific balance group. The adjustment amount is $9.50. The second record specifies a similar adjustment, except that this adjustment has no tax reversal. The last record specifies an adjustment of 10 free domestic minutes to the account's default balance group. BRM does not apply a tax reversal for this adjustment.

To run bulk adjustments, generate a CSV file and then run the "pin_apply_bulk_adjustment" utility to load the file into the database.

1. Generate the CSV file and check it for format problems and spurious blank lines.
2. Use the following command to run the **pin_apply_bulk_adjustment** utility:

   ```
   pin_apply_bulk_adjustment -f input_file.csv
   ```

   where *input_file* is the file name and path of your input file.
3. Check the **pin_bulk_adjust.pinlog** file for any failed adjustments.

   The bulk adjustment application generates an intermediate file in flist format. The default name of this file is **pin_mta_search.flist**.

   All of the CSV records that failed to process due to the wrong format or server-side errors are written to an error file. The default name of this file is **pin_mta_failed.csv**.

   The names and locations of these files are defined in the **pin_apply_bulk_adjustment pin.conf** file.

3

Opening and Resolving Disputes

Learn how to open and resolve disputes in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

About Disputes and Settlements

A CSR creates a dispute when customers disagree with the amount they are asked to pay and the problem requires investigation before it can be resolved. Disputes and settlements credit or debit the currency or noncurrency balances of a customer's account but do not return money to the customer directly.

The dispute process involves two distinct activities: opening a dispute and settling that dispute. A dispute is a transaction that records a customer's objection to a currency amount billed to an account. A settlement is a transaction that resolves a dispute by crediting or debiting all, part, or none of the dispute amount to the account.

As with payments and adjustments, the BRM system creates a dispute item when the CSR enters a dispute. The disputed amount typically reduces the Due of a bill and the current balances of the customer's account. BRM does not request payment for the disputed amount. If an item is under dispute (that is, the disputed field is nonzero) the bill item remains open even if the Due becomes zero.

In settling a dispute, the CSR can grant the entire disputed amount to the customer, grant only part of the disputed amount, or deny the dispute and again request payment for the entire disputed amount. The BRM system creates a settlement item for the amount that is not granted and transfers amounts between the settlement item and the disputed bill item. After the dispute resolution, the Due and the current balance in the customer's account balance group reflect what the customer owes.

When the dispute item is created, its Total and Due are initially equal, but the BRM system immediately transfers the credit in the Due field of the dispute item to the disputed bill item, makes the Due of the dispute item zero, and closes the dispute item.

After a settlement, the amount of the dispute that is granted to the customer appears in the Adjusted field of the disputed bill item and in the histories of the disputed item and the settlement item.

For currency balances, credit disputes decrease the Due of a bill or bill item. If the customer is disputing the currency balance of an event for credit, the dispute removes the balance impact of the disputed event from the account. Debit disputes have the opposite effect.

Note

For noncurrency balances, event disputes are recorded but do not directly affect the balance until the dispute is settled. At that point, the disputed balance is increased by the amount of the settlement.

Disputes and settlements are made at various levels, as appropriate to the situation. The way that BRM processes disputes and settlements varies slightly from level to level, as follows:

- **Bill level**: CSRs can dispute or settle an entire bill or a selection of bill items. In either case, BRM creates a single dispute or settlement item and alters the Due of each bill item covered by the dispute or settlement. Bill disputes and settlements can be thought of as a set of item disputes or settlements covered under the umbrella of a single dispute or settlement item.

  Bill disputes and settlements can act against A/R bills only. CSRs cannot dispute or settle a bill from a nonpaying bill unit by filing a bill dispute or settlement directly against that bill. Instead, they file the bill dispute or settlement against the parent A/R bill. The dispute amount cannot exceed the total amount of the bill against which the dispute is applied, and the settlement amount cannot exceed the total dispute amount for the bill.
- **Item level**: Disputes and settlements at this level operate as either a CSR-initiated action against a single bill item or as a unilateral action against a set of bill items initiated by a bill dispute or settlement. Depending on how the dispute or settlement was initiated, BRM creates dispute and settlement items as follows:
  - If the CSR initiated the dispute or settlement at the item level, one dispute or settlement item is created for the item he or she chooses.
  - If the dispute or settlement was initiated at the bill level, a single dispute or settlement item will cover all the individual item disputes that make up the bill dispute.

  To customize item level disputes, use these opcodes:
  - PCM_OP_BILL_POL_VALID_DISPUTE. See "Customizing Item Disputes" in *BRM Opcode Guide*.
  - PCM_OP_BILL_POL_VALID_SETTLEMENT. See "Settling Disputed Items" in *BRM Opcode Guide*.
- **Event level**: CSRs can dispute and settle any event and they can dispute or settle multiple events from an account in one operation.

  When an event dispute is opened, BRM creates one dispute event for each disputed event, establishing a one-to-one correspondence between the dispute event and the original event. The dispute event updates the original item's Dispute field. BRM bundles all the individual dispute events into one dispute item.

  Similarly, when you settle an event dispute, BRM creates a settlement event for each dispute event. In this case, BRM transfers the settlement amount to the Adjusted field of the original item and the denied amount to the Due field. As with disputes, BRM bundles all individual settlement events into one settlement item.

Table 3-1 summarizes the dispute and settlement types.

Table 3-1 Dispute and Settlement Types

Dispute or Settlement Type	Currency	Noncurrency
Bill dispute or settlement	Yes	No
Item dispute or settlement	Yes	No
Event dispute or settlement	Yes	Yes

CSRs open and resolve disputes by using Billing Care or Customer Center.

Reserving and Freeing Balances for Disputes and Settlements

To ensure that balances are not misused when a dispute is open, BRM reserves balances disputed at the event level. The reservation process protects the account balance from being credited for the dispute amount until settlement occurs.

For example, if the customer disputed a 10-minute charge on his bill, this amount would be credited to the account balance and, for prepaid accounts, would increase the prepaid amount. However, placing a reservation on those 10 minutes lowers the credit limit for the account and prevents the disputed time from being freely used as it would be in a normal credit situation.

Configuring Event Notification for Disputes and Settlements

When an event dispute is opened or settled, BRM uses event notification to call opcodes that perform the appropriate follow-up operations. Disputes and settlements use the following events for event notification:

- **/event/billing/dispute/notify**
- **/event/billing/settlement/notify**

Before you can use the event dispute and settlement feature, you must configure the event notification feature as follows:

1. If your system has multiple configuration files for event notification, merge them.

   See "Merging Event Notification Lists" in *BRM Developer's Guide*.

2. Ensure that the merged file includes the following information from the *BRM_home*/**sys/data/config/pin_notify** file:

   ```
   # Settlements and disputes related event notifications
   2354    0    /event/billing/settlement/notify
   2355    0    /event/billing/dispute/notify
   ```

3. (Optional) If necessary to accommodate your business needs, add, modify, or delete entries in your final event notification list.

   See "Editing the Event Notification List" in *BRM Developer's Guide*.

4. (Optional) If necessary to accommodate your business needs, create custom code for the event notification to trigger.

   See "Triggering Custom Operations" in *BRM Developer's Guide*.

5. Load your final event notification list into the BRM database.

   See "Loading the Event Notification List" in *BRM Developer's Guide*.

For more information, see "Using Event Notification" in *BRM Developer's Guide*.

4 Configuring Reasons for Adjustments, Disputes, and Settlements

Learn how to define the reasons in reason codes, in Oracle Communications Billing and Revenue Management (BRM). When your CSRs enter an adjustment, dispute, or settlement, they can provide a reason for the action such as "Net Speed Was Slow".

Topics in this document:

Configuring Reason Codes for Adjustments, Disputes, and Settlements

Reason codes explain why an adjustment, dispute, or settlement is being granted. You typically select from a list of reasons appropriate to the action.

To implement reason codes, you create and load a **reasons**.*locale* file that lists each reason code and associates it with a reason code domain. The domain classifies the reason code according to the type of activity it applies to. You can also include information in the **reasons**.*locale* file that prevents inappropriate event adjustments based on the reason code and service type.

The **reasons**.*locale* file defines each reason code domain, the reason codes that belong to the domain, and the event G/L ID. The domain and reason code information is used to build the **/strings** object, and the event G/L ID is used to build the **/config/map_glid** object.

For event adjustments, you can also use the service block to define which services are valid for the reason code and specify the valid event types within those services. The service block is optional; it is used to define the reason code scope. If you do not include it, BRM adjusts all events in the batch regardless of whether the reason for the adjustment makes sense for the service and event.

The following example shows a single **reasons**.*locale* file segment defining a reason code domain.

```
DOMAIN = "Reason Codes - Event Adjustment Reasons";
STR
  ID = 3;
  VERSION = 22;
  STRING = "Net Speed Was Slow";
  SERVICE
    TYPE = "/service/ip";
    EVENT
      TYPE = "/event/session";
    EVENT-END
    EVENT
      TYPE = "/event/session/call";
    EVENT-END
  SERVICE-END
  SERVICE
    TYPE = "/service/telco/gsm/data";
    EVENT
      TYPE = "/event/session/gsm/telco";
    EVENT-END
  SERVICE-END
  EVENT-GLID
    "/event/billing/adjustment/event"         105;
  EVENT-GLID END
END
```

For more information about the **reasons**.*locale* file, see "String Manipulation Functions" in *BRM Developer's Reference*.

To define reason codes for adjustments, disputes and settlements, you edit the **reasons.en_US** sample file in the *BRM_home***/sys/msgs/reasoncodes** directory. You then use the **load_localized_strings** utility to load the contents of the file into the **/strings**, **/config/map_glid**, and **/config/reason_code_scope** objects. See "load_localized_strings" in *BRM Developer's Guide* for more information.

When you run the **load_localized_strings** utility, use this command:

```
load_localized_strings reasons.locale
```

Note

- If you are loading a localized version of this file, use the correct file extension for your locale.
- If you add your own reason codes to the **reasons**.*locale* file, use IDs above 100,000.

Caution

The **load_localized_strings** utility overwrites existing **/config/map_glid** and **/config/reason_code_scope** objects. If you are updating these objects, you cannot load new G/L ID maps and reason code only. You must load complete sets of data each time you run the **load_localized_strings** utility. This is also true when using the **/strings** object, but only if you specify the **-f** parameter. Otherwise, the **load_localized_strings** utility appends the new data to the object.

5

Giving Refunds to Customers

Learn how to create refunds to return money that your company owes to its customers in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

About Refunds

You can give customers a refund whenever your company owes them money. Unlike an adjustment, which credits the customer's account balances, a refund returns money that your company owes a customer directly to the customer.

Refunds for BRM-initiated payments return money to the direct debit or credit card account the customer uses to make payments. For customers who pay invoices, your company makes a refund by check or other externally initiated payment method.

Note

You cannot refund suspended payments.

Giving refunds to customers is a two-step process. You first create refunds as refund items in the BRM database. You then run the **pin_refund** utility to deposit the refunds and close the refund items.

There are two ways to create refund items:

- Use Billing Care or Customer Center to create refund items for individual accounts. You can create refunds for accounts or bills that have a credit balance.
- Use the **pin_mass_refund** utility to create refund items for all accounts that have a credit balance (that is, all accounts that your company owes money).

When creating a refund item, the BRM evaluates the open bill items and A/R action items for the account, transfers credit among the items to close as many as possible, creates a refund item to contain the remaining credit, and closes the remaining open items that had a credit in Due.

When you refund an account whose bill unit is nonpaying, the refund is applied to the account that owns the paying parent bill unit. BRM associates the refund item with the parent bill unit's balance group that has the greatest amount due. If the parent bill unit has no amount due, the refund item is associated with the default balance group of the account that owns the parent bill unit. For more information about bill unit hierarchies, see "Managing Account and Bill Unit Hierarchies" in *BRM Managing Customers*.

Creating Refund Items for All Accounts with a Credit Balance

Run the **pin_mass_refund** utility to create a refund item for accounts that have a credit balance. See "pin_mass_refund".

Include the **pin_mass_refund** utility in any of the billing scripts, such as **pin_bill_day**. Run the **pin_mass_refund** utility before you run the **pin_refund** utility. See "Running the Billing Scripts" in *BRM Configuring and Running Billing*.

Giving Refunds for BRM-Initiated Payments

Run the **pin_refund** utility to give refunds to all accounts that have open refund items. See "pin_refund".

You do not specify start and end date parameters when you run this utility. If you miss a billing day, the **pin_refund** utility processes all existing refund items.

The **pin_refund** utility is included by default in the **pin_bill_day**, **pin_bill_week**, and **pin_bill_month** scripts. For information about these billing scripts, see "Running the Billing Scripts" in *BRM Configuring and Running Billing*.

Note

When you use multiple clearing house vendors, you run this utility for each clearing house.

Giving Refunds for Externally Initiated Payments

To make refunds for externally initiated payments, you first create the refund items. You then make the refund payments by check or by an externally initiated payment method.

Create the payments outside of your BRM system and then manually record them in BRM.

Tip

You can create a custom application that finds refund items and sends the amount and account identification to a check-writing program.

Specifying the Minimum Amount to Refund

You can specify the minimum amount to give as a refund. The **pin_refund** billing utility processes refund items with an amount greater than the minimum you specify.

The minimum value is expressed in terms of the account currency.

To specify the minimum amount to refund:

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

   ```
   pin_bus_params -r BusParamsAR bus_params_AR.xml
   ```

   This command creates an XML file named **bus_params_AR.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_AR.xml.out**, set **MinimumRefund** to the minimum amount to refund:

   ```
   <MinimumRefund>value</MinimumRefund>
   ```

   The default value is **2.00**.

   > **Caution**
   >
   > BRM uses the XML in this file to overwrite the existing instance of the **/config/business_params** object. If you delete or modify any other parameters in the file, these changes affect the associated aspects of the BRM configuration.

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

   ```
   pin_bus_params bus_params_AR.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*.

You do not need to restart the Connection Manager (CM) to update this entry.

Defining Non-refundable Items

By default, only refund items are non-refundable. To make other credit items non-refundable, run the **pin_bus_params** utility to change the **NonrefundableCreditItems** business parameter. See "pin_bus_params" in *BRM Developer's Guide* for more information.

To add non-refundable item types:

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

   ```
   pin_bus_params -r BusParamsAR bus_params_AR.xml
   ```

3. In the XML file, add the non-refundable **/item** objects after the **/item/refund** entry, separated by commas:

```
<NonrefundableCreditItems>/item/refund</NonrefundableCreditItems>
```

> **Note**
>
> Do not remove **/item/refund**.

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

```
pin_bus_params bus_params_AR.xml
```

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

Reversing Refunds

If a refund fails, you can reverse the refund. Refunds can fail because of returned checks and invalid credit cards.

> **Note**
>
> - You can reverse only failed refunds.
> - A refund can be reversed only once. You cannot reverse a refund that has been reversed before.

To reverse a refund, write a custom application that calls the PCM_OP_AR_REVERSE_REFUND opcode. See "Reversing Refunds" in *BRM Opcode Guide*.

6

Configuring Write-Offs

Learn how to perform write-offs in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

About Write-Offs

> **Note**
>
> - Before you write off an account or bill unit (**/billinfo** object), ensure that all items have been billed and that no pending or unbilled items for the account or bill unit remain.
> - To ensure that you can reverse a write-off on an account, the account status must be **inactive** before you write off the account.

A write-off removes from your company's assets an A/R amount that your company has determined the customer will never pay. A write-off can also remove an A/R amount that your company has decided it will not refund to the customer; for example, if the amount is very small or if you sent the customer a check that was returned because the customer moved without leaving a new address.

For write-offs, you *always* write off the entire account, bill, or bill item.

You can write off the following:

- **Accounts:** Write-offs are not available for an account that owns a top-level paying bill unit if its Due is zero. The account should be inactivated before it is written off.
- **Bills:** Write-offs are not available for a bill if it is pending or its Due is zero.
- **Bill items:** Item-level write-offs are allowed for pending and open items.

The following types of events are created for a write-off:

- A write-off event holding the total net amount of the write-off with a G/L ID for the net amount
- A tax write-off event holding the VAT amount of the write-off with a G/L ID for tax amount

To perform a write-off:

- The items to write off must be open.
- The write-off amount must be less than or equal to the amount due.

BRM creates a write-off item for the amount to be written off. It transfers the Due from the write-off item to the bill items being written off and closes the bill items.

Writing off uncollectable debt or an unpayable credit lets you control how it is treated in your accounting and G/L reporting system. Depending on how your company has set up its G/L system, the amount written off is transferred from A/R to a bad debt account or, for unpayable credit, to a miscellaneous revenue account.

A write-off always includes taxes, but you can specify whether the net and tax amounts are written off in one or two events. If you write off in one event, the net and tax amounts are stored separately within the event and can be mapped to different sets of G/L accounts. If you write off in two events, one event stores the net amount of the write-off while another event stores the tax amount. The net and tax amounts stored in these two events can also be mapped to separate sets of G/L accounts. However, the way you specify mapping information for G/L accounts is different for the one-event and the two-event write-off.

You can use the PCM_OP_BILL_POL_VALID_WRITEOFF to customize how items are validated for a write-off. See "Customizing Write-Off Validation" in *BRM Opcode Guide*.

7

Configuring Write-Off Reversals

Learn how to perform write-off reversals in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

About Write-Off Reversals

Write-off reversals enable you to re-allocate payments to accounts and bill units that were written off due to unpaid balances. Unpaid bills and item charges are generally written off as unrecoverable debt, but this feature enables the amount written off to be recovered and allocated to open bills and items of an account.

The following steps occur when a payment is made for a written-off amount. It also describes the steps that occur if the original payment must be reversed and the paid amount must be written off again:

1. An account or bill unit contains an unpaid balance that was written off as unrecoverable, and a payment is received for the written-off amount.
2. If the write-off reversal feature is enabled, the write-off on the account or bill unit is reversed.
3. The written-off balance is transferred back to the account or bill unit.
4. The payment is applied to the account or bill unit and all paid-off bill items are closed.
5. Any remaining balance on the account or bill unit (for an underpayment) is written off again.
6. If the payment fails, a reversal of the payment is submitted.
7. BRM verifies that the original payment was applied to the written-off account or bill unit.
8. The second write-off on the account or bill unit is reversed, if one exists (when the write-off reversal payment was an underpayment).
9. Any bill items closed by the original payment are reopened and the amount is transferred back to the account or bill unit.
10. The original payment is reversed.
11. After the payment reversal, if the account or bill unit has a balance due, the amount is written off again to restore the account balance to **0**, and the account or bill unit status is reset to write-off.

About Overpayment and Underpayment Allocation to a Written-off Account or Bill

After a payment is posted to a written-off account or bill unit, the handling of the bill or bill items is determined by the amount of the payment: exact payment, overpayment, or underpayment. Exact payments are allocated to the appropriate bill and bill items, which are closed after being paid, and the account's write-off flag is *not* reset to write-off.

Overpayments and underpayments are allocated according to the rules defined in your business policies:

- In general, if the payment amount is *more than* the amount written off, the paid bills and bill items are closed and the remaining amount is left unallocated in the account. The account or bill unit is left open and the account's write-off flag is *not* reset to write-off.
- If the payment amount is *less than* the amount written off, the payment is fully allocated, and the open bills and bill items are closed again. The remaining amount is written off and the account's write-off flag is reset to write-off.
- If an underpayment fails and you reverse it, both the amount that was written off (due to the underpayment) and the amount that was removed by the payment are returned to the account or bill unit. This restores the account balance to its original state so the entire amount can be written off again. This corresponds to the initial write-off amount for the account or bill unit.

  For example, if your company writes off a $50 unpaid balance on an account and 6 months later receives a payment of $45 for that account, the $50 write-off is returned to the account so that the payment can be applied toward the balance. After the payment amount is subtracted, the remaining $5 balance is written off again.

  Later, if the $45 payment fails, the following steps occur when you reverse the payment:

  1. The $5 write-off amount is returned to the account balance.
  2. The $45 balance that was removed by the initial payment is returned to the account balance.
  3. The $50 account balance is written off again.
- If any open unallocated items are in the account at the time of the reversal, the re–write-off on the account does not occur. You must first allocate and close the open items, or customize the PCM_OP_BILL_POL_REVERSE_PAYMENT policy opcode to allocate and close the open items, before performing the reversal.

Defining Reason Codes for Write-Off Reversals

You define additional reason codes in the **reasons**.*locale* file for the write-off reversal event and the re-write-off of an account. You use the Credit Reasons domain (version 8) for write-off reversal reasons, and you should map the reason code to the **/event/billing/writeoff** G/L ID, as shown in the following sample:

```
DOMAIN = "Reason Codes-Credit Reasons" ;
STR
     ID = 4 ;
     VERSION = 8 ;
     STRING = "Write-off for Auto-writeoff reversal feature" ;
     EVENT-GLID
       "/event/billing/writeoff"     110;
     EVENT-GLID-END
END
```

> **Note**
>
> If you add your own reason codes to the **reasons.***locale* file, you should use IDs above 100,000.

To define reason codes for write-off reversals, you edit the **reasons.en_US** sample file in the *BRM_home***/sys/msgs/reasoncodes** directory. You then use the **load_localized_strings** utility (see "load_localized_strings" in *BRM Developer's Guide*) to load the contents of the file into the **/strings** objects.

When you run the **load_localized_strings** utility, use the following command:

```
load_localized_strings reasons.locale
```

> **Note**
>
> If you are loading a localized version of this file, use the correct file extension for your locale.

Mapping G/L IDs to Write-Off Reversal Events

You can track funds you recover from write-off reversals in your general ledger (G/L) system. In addition, you can track the following events that occur if the payment fails and must be reversed:

- If the payment was an underpayment, the write-off reversal event that occurred for the second write-off, which was required to write off the unpaid balance on the account.
- The reversal of the original payment that failed.
- The re-write-off of the debt on the account.

Depending on your G/L configuration, you can create single or separate G/L entries to track the reversed written-off amount and associated tax amount in respective G/L accounts.

You map custom G/L IDs for write-off reversal events by editing and loading the **reasons.***locale* file. This file contains the reason code definition that is assigned to the G/L ID of the **/event/billing/writeoff** event.

To load a customized **reasons.***locale* file into the BRM database, use the **load_localized_strings** utility. This utility loads the event-to-G/L ID mapping into a **/config/map_glid** object in BRM. The G/L ID-to-event mapping is defined in the **pin_glid** file.

You map custom G/L IDs when you define your reason codes for the write-off reversal event. See "Defining Reason Codes for Write-Off Reversals".

For more information about general ledger data, see "About Collecting General Ledger Data" in *BRM Collecting General Ledger Data*.

Enabling Automatic Write-Off Reversals during Payment Collection

You can configure BRM to reverse write-offs on accounts and bill units automatically when payments are received for written-off amounts. This enables the payment to be posted to the accounts and bill units and the written-off amount to be recovered immediately during payment processing.

To enable this feature, run the **pin_bus_params** utility to change the **AutoWriteOffReversal** business parameter. For information about this utility, see "pin_bus_params" in *BRM Developer's Guide*.

To enable automatic write-off reversals during payment collection:

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

   ```
   pin_bus_params -r BusParamsAR bus_params_AR.xml
   ```
3. In the XML file, change **disabled** to **enabled**:

   ```
   <AutoWriteOffReversal>enabled</AutoWriteOffReversal>
   ```
4. Save the file as **bus_params_AR.xml**.
5. Load the XML file into the BRM database:

   ```
   pin_bus_params bus_params_AR.xml
   ```
6. Stop and restart the CM.

For information on how to customize the write-off level to search for and to be available for reversal during payment processing, see "Customizing Write-Off Reversals" in *BRM Opcode Guide*.

Enabling Write-Off Reversals for Bill Units

By default, BRM is configured to reverse write-offs for accounts when payments are received.

You can configure BRM to reverse write-offs for bill units when payments are received for a written-off bill unit. When you configure BRM to perform write-off reversals for bill units, BRM reverses write-offs for both bill units and accounts.

To enable this feature, run the **pin_bus_params** utility to change the **WriteOffLevel** business parameter. For information on this utility, see "pin_bus_params" in *BRM Developer's Guide*.

To enable write-off reversals for bill units:

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

   ```
   pin_bus_params -r BusParamsAR bus_params_AR.xml
   ```
3. In the XML file, change **account** to **billinfo**:

   ```
   <WriteOffLevel>billinfo</WriteOffLevel>
   ```
4. Save the file as **bus_params_AR.xml**.
5. Load the XML file into the BRM database:

```
pin_bus_params bus_params_AR.xml
```

6. Stop and restart the CM.

8 About Transferring Rollover Balances

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

Topics in this document:

About Allowing Customers to Transfer Rolled-Over Balances

Balances rolled over from one billing cycle to another are usually rolled over within the same account. By using the rollover-transfer feature, you can transfer noncurrency rollover balances from one account or service instance to other accounts or service instances.

BRM can perform rollover transfers in the following ways:

- From one account to other accounts.
- From a service instance in one account to service instances in other accounts.
- From one account to service instances in other accounts.
- From a service instance in one account to other accounts.
- From a service instance in one account to other service instances in the same account.
- Between services, even when the service types differ.

  The sender's service type does not need to match the receiver's service type as long as both services own the same type of balance, such as dollars or minutes.

BRM cannot transfer rolled over balances in the following circumstances:

- A member service shares a balance group with its subscription service.
- The buckets were created as a result of a previous rollover transfer. Previously rolled over balances cannot be rolled over again.

If you offer subscription services, you can transfer rolled-over balances from a subscription service to a member service only if they belong to different balance groups. A member service that shares the same balance group as its subscription service *can receive* a rollover transfer but *cannot send* a rollover transfer.

About the Rollover-Transfer Profile

You specify how to transfer rollovers by using a rollover-transfer profile. The rollover-transfer profile specifies the following:

- The account or service that owns the balance grant (the sender)

- The list of balances to roll over
- The accounts or services into which the rollover is transferred (the receiver)
- (Optional) The validity periods in which the rollovers can transfer

You create, modify, and delete rollover-transfer profiles in Customer Center.

When you create a rollover-transfer profile, BRM, by default, verifies that the profile does not have overlapping validity periods or receivers for a given service. You can add or modify the verification checks by customizing the PCM_OP_CUST_POL_VALID_PROFILE policy opcode.

You can select any noncurrency balance to configure a rollover transfer unless there is a rollover rule defined in a charge offer and the sender has not purchased that charge offer. The sender must own the charge offer in which the rollover rule is defined for the rollover transfer to occur.

About Rerating the Receiver's Account Due to Delayed Billing

When delayed billing is configured, balances rolled over to the receiver during the initial billing process can be accessed by delayed events of the sender.

After a rollover transfer, if the sender account is rerated and the rollover buckets are adjusted, the receiver's rollover amount is also affected. When a bill is finalized for the sender, BRM generates a rollover correction event to adjust the rollover of the sender if any rolled-over balances were consumed by delayed events. BRM adjusts the rollover transfer amount of the receiver accordingly.

Note

- You must manually rerate the receiver's account if events rated before the rollover correction were consumed from the rollover-transfer amount. BRM does not automatically create rerate jobs for the receiver.
- The receiver must be rerated if there is rollover correction for the sender, even when the sender is not rerated.

Configuring BRM to Use Rollover Transfers

Before you can use rollover transfers, you must perform the following tasks:

1. Enable rollover transfers in the **/config/business_params** object.
2. Set up event notification for rollover transfers.

To enable rollover transfers, run the **pin_bus_params** utility to change the **RolloverTransfer** business parameter. See "pin_bus_params" in *BRM Developer's Guide*.

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

   ```
   pin_bus_params -r -c "Subscription" bus_params_subscription.xml
   ```

3. In the XML file, change **disabled** to **enabled**:

   ```
   <RolloverTransfer>enabled</RolloverTransfer>
   ```

4. Save the file as **bus_params_subscription.xml**.

5. Load the XML file into the BRM database:

   ```
   pin_bus_params bus_params_subscription.xml
   ```

6. Stop and restart the CM.

To configure event notification for rollover transfers, do the following:

1. Depending on which BRM features you use, your system may contain one or more configuration files for event notification.

   If your system contains more than one of these files, you must merge their contents into a single file.

   All of the event notification files available in your system are in the *BRM_home***/sys/data/config** directory.

2. Add the following entries to the event notification file:

   ```
   # Event notification for rollover transfers
   9069    0    /event/billing/cycle/rollover/monthly
   9069    0    /event/billing/cycle/rollover_correction
   ```

   This configures the **/event/billing/cycle/rollover/monthly** and **/event/billing/cycle/rollover_correction** events to call the PCM_OP_SUBSCRIPTION_TRANSFER_ROLLOVER opcode (opcode ID number 9069) to determine if a rollover transfer can occur.

3. (Optional) If necessary, add, modify, or delete entries in your event notification file.

4. (Optional) If necessary, create custom code for event notification to trigger.

5. Load your event notification file into the BRM database's **/config/notify** object by running the **load_pin_notify** utility. See "load_pin_notify" in *BRM Managing Customers*.

For more information, see "Using Event Notification" in *BRM Developer's Guide*.

9

Managing Balances

Learn how Oracle Communications Billing and Revenue Management (BRM) stores, displays, and manipulates balance data.

Topics in this document:

Transferring Balance Amounts

Transfers are made to correct payments that were allocated incorrectly, to allocate a debit adjustment for any account, and to allocate an account adjustment for an account that pays by credit card. In Customer Center and Billing Care, a CSR can transfer amounts between bill items. Transfers can also be made from payment items and account adjustment items to bill items.

A CSR can also make a credit or debit transfer of a currency balance.

Moving a Balance Group from One Bill Unit to Another

To move a balance group from one bill unit to another in the same account, use Billing Care or Customer Center.

Moving a balance group to a different bill unit means that any new charges for the services in the balance group are applied to the new bill unit.

For example, an account has two bill units. One bill unit tracks charges for services that are invoiced. The other tracks charges for services paid by credit card. The customer decides to pay for all services by credit card. The balance group for invoiced services is moved to the bill unit with the credit card payment method. Any new charges for these services are applied to the new bill unit and are charged to the credit card. Existing charges for these services that occurred before the balance group was moved are associated with the old bill unit and are invoiced.

You cannot move a balance group to another bill unit if the balance group's bill unit has unallocated payments or adjustments, open refunds, or unresolved disputes. All disputes must be settled, refunds paid, and payments and adjustments allocated before the balance group can be moved.

A bill unit must have at least one balance group. When a bill unit has only one balance group and that balance group is moved to another bill unit, BRM automatically creates a new balance group not associated with any service for the bill unit from which the balance group was moved.

Important:

- You cannot move account default balance groups to a different bill unit.
- You should never move a balance group to a bill unit in a different account. To have a different account be responsible for the charges in a balance group, you must create a bill unit hierarchy that includes the appropriate bill units from both accounts.

> **Note**
>
> To move a balance group from one bill unit to another in the same account, create a custom application that calls PCM_OP_CUST_MODIFY_CUSTOMER.

Backdating A/R Actions

You can backdate A/R actions so that, for accounting purposes, the action is considered to have occurred at an earlier point in time and the revenue for that time is reported accurately. The A/R actions you can backdate include adjustments, disputes, settlements, externally initiated payments, payment reversals, and write-offs.

> **Note**
>
> BRM does not support future dating of A/R actions.

To backdate an A/R action, the CSR sets the transaction date (the date that the action is to take effect) to a date before the current date. When backdating, the CSR must select a date after the following:

- The last posted G/L transaction report.
- The account creation date.

Whether backdated A/R actions appear on the current bill depends on the accounting method you use:

- If you use balance forward accounting, the total of A/R actions for all prior open bills appears on the current bill or invoice as part of the previous balance.
- If you use open item accounting, A/R actions do not appear on the current bill or invoice.

If customers request an invoice that includes information on specific A/R actions in a prior billing period, you can create the invoice by running PCM_OP_INV_MAKE_INVOICE.

Synchronizing BRM and ECE Balance Group Transfer Data

You must configure BRM to synchronize the following database updates with the ECE cache:

- Service transfers to a different balance group
- Balance group transfers to a different bill unit

To configure BRM to send updated balance group transfer information to ECE, use the account synchronization feature to publish the **ServiceBalanceGroupTransfer** business event to the Oracle DM database queue. For installation and configuration instructions, see "Installing and Configuring Account Synchronization" in *BRM Installation Guide*.

When configuring the Oracle DM for account synchronization:

- Make sure the **ServiceBalanceGroupTransfer** business event is listed in your payload configuration file under the **<PublisherDefs>** section. This business event appears in the default Oracle DM payload configuration file (*BRM_home*/**sys/eai_js/payloadconfig_ifw_sync.xml**).
- Make sure the event notification list maps the **/event/notification/service_balgrp_transfer/data** notification event to opcode number **3626**. This mapping appears in the default account synchronization event notification file (*BRM_home*/**sys/data/config/pin_notify_ifw_sync**).
- Add the **ServiceBalanceGroupTransfer** business event to your *BRM_home*/**sys/dm_oracle/ifw_sync_queuenames** file if the file maps specific business events to queues.

10

Accounts Receivable Utilities

Learn about the syntax and parameters used by the Oracle Communications Billing and Revenue Management (BRM) accounts receivable utilities.

Topics in this document:

pin_apply_bulk_adjustment

Use this BRM command-line utility to apply an adjustment across a broad range of accounts, such as all accounts charged a regular day-time rate when they should have been charged a holiday rate. See "Adjusting Multiple Accounts Simultaneously".

Location

BRM_home/**bin**

Syntax

```
pin_apply_bulk_adjustment [-verbose file_name.log] [-test] [-h] -f input_file.csv
```

Parameters

-verbose
Displays information about successful or failed processing as the utility runs.

-h
Displays the utility's syntax and parameters.

-f *input_file*.csv
Specifies the name and location of the file that specifies how BRM will apply the bulk adjustment; for example, **C:\bulkadj\bulk_run_1.csv**.

Results

Each account adjustment initiated during a bulk adjustment is handled as a separate transaction. Therefore, if any single adjustment in the bulk adjustment fails, BRM does not need to roll back the entire bulk adjustment. Rather, the adjustment remains in place for all accounts that had successful adjustments, and the utility identifies the accounts that failed to adjust so you can correct the problem. The utility provides this feedback in a log file (**default.pinlog**) that it generates in its start directory or in a directory specified by the configuration file.

pin_mass_refund

Use this utility to create a refund for accounts that have a credit balance. See "Giving Refunds to Customers".

Note

For multischema systems, you must run the utility separately against each database schema in your system. See "Running Non-MTA Utilities in Multischema Systems" in *BRM System Administrator's Guide*.

Location

BRM_home/**bin**

Syntax

```
pin_mass_refund   -active|-close|-inactive
                  [-pay_type payment_method]
                  [-test] [-verbose file_name.log] [-test] [-help]
```

Parameters

-active | -close | -inactive
Specifies the status of the accounts for which to create refund items.

-pay_type *payment_method*
Specifies the payment method, for example:

- **10003** for credit card
- **10005** for direct debit
- **10018** for SEPA

-test
Finds the bill units (**/billinfo** objects) that meet the criteria but does not create any refund objects.

-verbose
Displays information about successful or failed processing as the utility runs.

-help
Displays syntax and parameters for this utility.

Results

If the utility does not notify you that it was successful, look in the utility log file (**default.pinlog**) to find any errors. The log file is either in the directory from which the utility was started, or in a directory specified in the configuration file.

pin_refund

Use this utility to find accounts that have refund items and make BRM-initiated refund payments to customers. See "Giving Refunds to Customers".

When you use multiple payment processors, you run this utility for each one. See "Using More Than One Payment Processor" in *BRM Configuring and Collecting Payments*.

Location

BRM_home/**bin**

Syntax

```
pin_refund   -pay_type payment_type_indicator
             [-vendor payment_processor_name]
             [-active | -close | -inactive]
             [-verbose file_name.log] [-test] [-help]
```

Parameters

-pay_type *payment_type_indicator*
Specifies the payment type, for example:

- **10003** for credit card
- **10005** for direct debit
- **10018** for SEPA

-vendor *payment_processor_name*
Specifies the credit card processor or automated clearing house (ACH) to use for validating credit cards, debit cards, and direct debit transactions.
This parameter is not applicable for the SEPA payment type.
For information on configuring payment processor information, see *BRM Configuring and Collecting Payments*.

-active | -close | -inactive
Specifies the status of the accounts to give refunds to.

-verbose
Displays information about successful or failed processing as the utility runs.

-test
Returns the number of refund items that would be created, but does not create any refund items.

-help
Displays syntax and parameters for this utility.

Results

If the utility does not notify you that it was successful, look in the utility log file (**default.pinlog**) to find any errors. The log file is either in the directory from which the utility was started, or in a directory specified in the configuration file.

When it is called internally by the **pin_bill_day** script, the **pin_refund** utility logs error information in the **pin_billd.pinlog** file.

pin_roll_up_ar_items

Use the **pin_roll_up_ar_items** utility to process temporary A/R items (**/tmp_ar_item_to_roll_up** object) of nonpaying child bill units and then roll up the balance impacts to the corresponding A/R items in the paying parent bill unit. You can run multiple threads of **pin_roll_up_ar_items** to process A/R items for different paying parent bill unit.

You must run this utility before billing the paying parent bill unit. For more information, see "Rolling A/R Actions Up to the Wholesale Parent" in *BRM Configuring and Running Billing*.

Note

To connect to the BRM database, the **pin_roll_up_ar_items** utility needs a configuration file in the directory from which you run the utility. See "Connecting BRM Utilities" in *BRM System Administrator's Guide*.

Location

*BRM_home***/bin**

Syntax

```
pin_roll_up_ar_items [-verbose] [-help]
```

Parameters

-verbose
Displays information about successful or failed processing as the utility runs.

-help
Displays syntax and parameters for this utility.

Results

If the **pin_roll_up_ar_items** utility does not notify you that it was successful, look in the utility log file (**default.pinlog**) to find any errors. The log file is either in the directory from which the utility was started or in a directory specified in the configuration file.

Error Handling

When the **pin_roll_up_ar_items** utility encounters an error while processing the A/R items in the temporary tables, it sets the PIN_FLD_BILLING_STATUS billing status field of the paying parent bill unit (**/billinfo** object) to PIN_BILL_ERROR. In addition, it sets the appropriate bit of the PIN_FLD_BILLING_STATUS_FLAGS field of the **/billinfo** object as PIN_BILL_FLAGS_UPDATE_ITEMS_ERROR (bit value 0x4000).

After you have resolved the processing errors, you can reprocess the A/R items by running the **pin_roll_up_ar_items** utility again.