Oracle® Communications Billing and Revenue Management

Suspending and Recycling Pipeline EDRs

Release 15.1

F93243-01

April 2025

Oracle Communications Billing and Revenue Management Suspending and Recycling Pipeline EDRs, Release 15.1

F93243-01

Contents

Preface

Audience vii
Documentation Accessibility vii
Diversity and Inclusion vii

1 About the EDR Recycling Features

About the EDR Recycling Features 1-1

2 About Standard Recycling

About Standard Recycling 2-1
About the Standard Recycling Workflow 2-1
About the Suspended EDR States 2-2
About the Standard Recycling Pipelines 2-3

3 Configuring Standard Recycling

About Configuring Standard Recycling 3-1
Configuring Pipeline Modules for Standard Recycling 3-2
Configuring a Preprocessing Pipeline 3-3
Configuring Standard Recycling in a Rating Pipeline 3-4
Configuring a Pre-Recycling Pipeline 3-6
Configuring Recycle Request Handling 3-7
Configuring a Pipeline Module to Add Recycle Keys to EDRs 3-8
Configuring SE Loader for Standard Recycling 3-8

4 Using Standard Recycling to Recycle Suspended EDRs

About Recycling Using pin_recycle 4-1
About the Standard Recycling Mechanism 4-1
Setting Up EDR Recycling by CDR File 4-2
About Recycling Suspended EDRs After Rating Interruptions 4-2
Setting Up EDR Recycling by Recycle Key 4-3

Setting Up pin_recycle to Run Periodically 4-3
Adding EDR Recycle Entries 4-3
Adding EDR Delete Entries 4-4

5 About Suspense Manager

About Suspense Manager 5-1
Suspending Individual CDRs, or CDRs in Bulk 5-2
Suspending CDR Files 5-3
Suspended Call Record States 5-4
About SE and SB Loaders 5-4
About the FCT_BatchSuspense Module 5-4
Differences Between the RE, SE, and SB Loaders 5-5
Suspense Manager Objects 5-5
About Upgrading from Standard Recycling to Suspense Manager 5-5

6 Installing and Configuring Suspense Manager

Installing Suspense Manager 6-1
About Configuring Suspense Manager 6-1
Planning and Setting up Your Database for Suspense Manager 6-3
Deciding Whether You Need to Extend the Suspense Subclasses 6-3
Selecting a List of Queryable EDR Fields 6-3
Adding /suspended_usage Subclasses with Queryable Fields 6-5
Creating a List of Editable Fields Based on Your /suspended_usage Subclasses 6-6
Loading Editable Fields into the Database 6-6
Changing the List of Suspense Reasons and Subreasons 6-6
Deciding Whether to Change the Suspense Reason and Subreason Lists 6-7
Changing the Suspense Reason and Subreason Lists 6-7
Configuring Pipeline Manager for Suspense Manager 6-9
Configuring FCT_PreSuspense 6-9
Configuring SuspenseCreateOutput 6-10
Configuring a Pre-Recycling Pipeline 6-10
Setting Up Suspended Event (SE) Loader for Suspense Manager 6-11
Setting Up Suspended Batch (SB) Loader for Suspense Manager 6-12
Creating Indexes for Search Templates 6-12
Configuring and Customizing Suspense Management Center 6-14
Adding Custom Fields to Suspense Management Center 6-14
Adding Custom Fields to Suspense Management Center Web Start 6-15
Configuring Event Notification for Suspense Manager 6-15
Configuring Debugging (Optional) 6-16
Configuring the Number of Suspended Records to Process in a Transaction 6-17

Suspense Management Center Permission Types 6-17

7 Customizing Suspense Manager

Processing a Large Number of Suspended Records 7-1
Overriding Pipeline Suspense Handling Rules 7-1
Changing the List of Override Reasons 7-2
Changing the List of CDR File Override Reasons 7-2
Using Suspense Management Center with Standard Recycling Call Records 7-2
About Suspense Manager Performance 7-3

8 Recycling EDRs in Pipeline-Only Systems

About Recycling EDRs 8-1
How the FCT_PreRecycle Module Works 8-2
How the FCT_Reject Module Works 8-3
How the FCT_Recycle Module Works 8-3
Recycling EDRs in Test Mode 8-4
Recycling EDRs 8-5
Configuring FCT_PreRecycle to Move Rejected EDRs to the Input Directory 8-5
Configuring FCT_Reject to Reject EDRs 8-6
Configuring FCT_Recycle to Recycle or Test Recycle EDRs 8-7

9 EDR Suspense Reasons

About Suspense Reasons 9-1
Pipeline Manager Suspense Reasons 9-1
DAT_Account Suspense Reasons 9-2
FCT_AggreGate Suspense Reasons 9-3
FCT_BalancePlugIn Suspense Reasons 9-3
FCT_CallAssembling Suspense Reasons 9-3
FCT_CustomerRating Suspense Reasons 9-4
FCT_Discount Suspense Reasons 9-4
FCT_EnhancedSplitting Suspense Reasons 9-5
FCT_ExchangeRate Suspense Reasons 9-5
FCT_MainRating Suspense Reasons 9-6
FCT_PreRating Suspense Reasons 9-6
FCT_Tadig2PlmnPlugIn Suspense Reasons 9-6
CIBER Suspense Reasons 9-7
Tap Suspense Reasons 9-7
FCT_AccountRouter Suspense Reasons 9-7
FCT_TriggerBill Suspense Reasons 9-7

FCT_Account Suspense Reasons 9-8
FCT_Filter_Set Suspense Reasons 9-8

10 Troubleshooting Suspense and Recycle Issues

Increasing Heap Size to Avoid Performance Problems 10-1
Unexpected Log Messages Caused by Missing MaxErrorRates Entry 10-2

Preface

This book describes how to correct and recycle event records that fail in Oracle Communications Billing and Revenue Management Pipeline Manager.

Audience

This guide is intended for pricing administrators and charging experts.

Documentation Accessibility

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

Access to Oracle Support

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

Diversity and Inclusion

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

1

About the EDR Recycling Features

Oracle Communications Billing and Revenue Management (BRM) includes several tools for managing suspended (failed) pipeline event data records (EDRs).

Topics in this document:

- About the EDR Recycling Features

About the EDR Recycling Features

BRM offers these tools for managing EDRs that Pipeline Manager does not successfully rate:

- **Standard recycling**. BRM provides the standard recycling tools as the default EDR recycling mechanism. You use the **pin_recycle** utility to test recycle, recycle, or delete EDRs that failed to process the first time through the pipeline.

  The standard recycling tools include:

  - The FCT_Reject pipeline module
  - The FCT_PreSuspense pipeline module
  - The FCT_Suspense pipeline module
  - The **pin_recycle** utility

  For details about standard recycling, see "About Standard Recycling".

- **Suspense Manager**. Suspense Manager is a service integration component that you purchase separately. It offers the most comprehensive and flexible set of tools for managing:

  - Individual failed CDRs
  - Large numbers of individual failed CDRs at once (bulk processing)
  - CDR files containing multiple CDRs (batch processing)

  Suspense Manager includes the Suspense Management Center GUI application to:

  - Analyze, edit, recycle, test recycle, write off, archive, and delete CDRs, either individually or in bulk.
  - Analyze, resubmit, write off, and delete batch files of CDRs.

  Suspense Manager also includes a set of BRM reports to analyze suspended call records. For details, see "About Suspense Manager".

- **Recycling EDRs for pipeline-only systems**. This feature is for systems that use Pipeline Manager but do not store suspended EDRs in the BRM database.

  This feature includes the FCT_Recycle and FCT_PreRecycle pipeline modules you use to recycle suspended EDRs. For details, see "Recycling EDRs in Pipeline-Only Systems".

2
About Standard Recycling

You use standard recycling to recycle, test recycle, or delete failed pipeline event data records (EDRs) in your Oracle Communications Billing and Revenue Management (BRM) system.

Topics in this document:

- About Standard Recycling
- About the Standard Recycling Workflow
- About the Suspended EDR States
- About the Standard Recycling Pipelines

About Standard Recycling

Standard recycling is the default recycling mechanism for Pipeline Manager event data records (EDRs). You use it to recycle, test recycle, or delete failed EDRs in your BRM system.

Standard recycling mainly relies on these BRM tools:

- The FCT_Reject pipeline module
- The FCT_PreSuspense pipeline module
- The FCT_Suspense pipeline module
- The Suspended Event (SE) Loader application
- The **pin_recycle** utility

You use the **pin_recycle** utility to recycle, test recycle, or delete suspended call records. EDRs are often suspended because of a pipeline configuration problem. You fix the problem and test recycle a CDR file of suspended call records. If they pass the recycle test, you can recycle all of the CDR files of suspended calls. The utility also has a delete option to remove call records that have been successfully processed or cannot be rated. For information, see "pin_recycle" in *BRM Pipeline Manager Reference*.

About the Standard Recycling Workflow

The standard recycling workflow consists of the following steps:

1. You start the pipeline with active FCT_PreSuspense, FCT_Suspense, and FCT_Reject modules.
2. FCT_PreSuspense appends suspense-related information to all EDRs that come through the pipeline.
3. As an EDR is processed, a module finds an error in the EDR. The module appends the error to the EDR and sets a flag to indicate that the EDR has an error.
4. The EDR is sent to the next module. Each module adds an error if any more are found.
5. The FCT_Reject module analyzes an EDR's errors to determine whether it has failed. It also routes EDRs to the appropriate output stream to be stored in the database by

Suspended Event (SE) Loader. SE Loader stores suspended EDRs in **/suspended_usage** objects.

By default, FCT_Reject fails call records with an error level of **Warning** or **Error**. However, you configure the error level or other conditions that cause EDRs to fail. Call records also "fail" if the pipeline cannot otherwise process them. These failures can be intentional or inadvertent. For example:

- A call record may arrive with invalid data and fail a Pipeline Manager validity rule.
- The call record may fail custom validity checking set up in a custom iScript.
- The Pipeline Manager database tables may be set up incorrectly.

6. During recycling operations, FCT_Suspense routes EDRs from **SuspenseCreateOutput** to **SuspenseUpdateOutput**.
7. You examine the errors and determine how to reconfigure Pipeline Manager to prevent the errors.
8. Run the **pin_recycle** utility with the **-f** *filename* option to start the recycling process. The utility sends the rejected EDRs through the pipeline again for another attempt to rate them.

   pin_recycle can recycle EDRs in test mode or real mode. Typically, you run the recycling processes in test mode first to see if the problems causing the EDR errors have been fixed. When there are no longer any errors, you recycle in real mode.

   You usually run **pin_recycle** (as part of a **cron** job) periodically.

   - In test mode, this utility creates a report about the processing but does not make any state changes.
   - In recycle mode, this utility sends the results to an output file and attaches a sequence number to it.

   > **Note:**
   >
   > This utility sends an entire CDR file to the error directory. You can configure the threshold for the number of errors allowed per file. See "Specifying the Maximum Errors Allowed in an Input File" in *BRM Setting Up Pipeline Rating and Discounting*.

9. Run **pin_recycle** again with the delete option to remove any remaining EDRs. For details on the **pin_recycle** utility, see "pin_recycle" in *BRM Pipeline Manager Reference*.

For details about configuring Pipeline Manager to use standard recycling, see "Configuring Standard Recycling".

For details about using standard recycling to recycle and delete EDRs, see "Using Standard Recycling to Recycle Suspended EDRs".

About the Suspended EDR States

The standard recycling process assigns suspended EDRs to one of the following states:

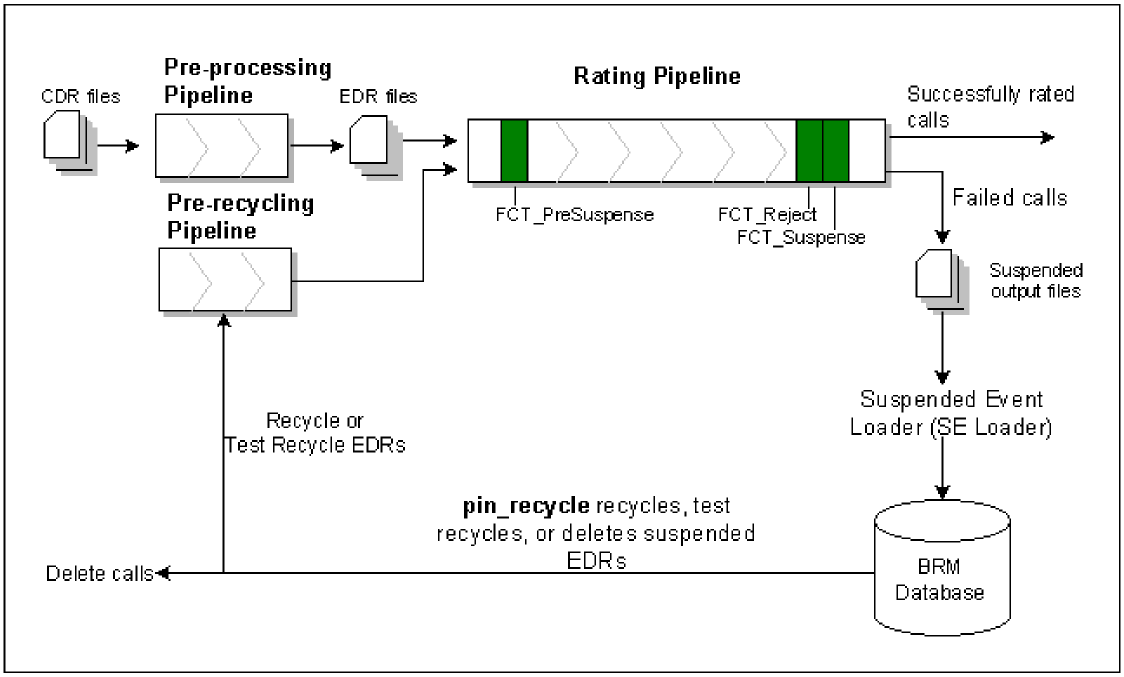
- **Suspended**: The call record could not be processed by the pipeline and has been stored in the BRM database as a suspended call record.
- **Recycling**: The call record is being sent through the rating pipeline again to be rated.
- **Succeeded**: The call record has been successfully recycled and rated.

- **Written off**: The EDR is set to this state automatically just before being deleted to generate revenue assurance data.

About the Standard Recycling Pipelines

Figure 2-1 shows how standard recycling fits into your BRM system.

Figure 2-1 Standard Recycling in BRM



The BRM and standard recycling processes consist of the following:

1. Call records first enter standard recycling through the preprocessing pipeline.
2. The *preprocessing pipeline* converts call records (CDRs) to EDRs used by BRM. Calls only go through this pipeline once, so only a few modules are appropriate, such as FCT_DuplicateCheck and FCT_CallAssemblings.
3. The *rating pipeline* is a normal rating pipeline. Most of your pipeline function modules are included in this pipeline. It is in this pipeline where you configure call "success" and "failure" policies. If calls "fail" in this pipeline, they are sent to a Suspended Event Loader (SE Loader).
4. *SE Loader* converts the failed calls to **/suspended_usage** objects in the BRM database.
5. You check your Pipeline Manager log files to see what caused the calls to fail.
6. You fix any errors, such as by reconfiguring the pipeline.
7. You run the **pin_recycle** utility to send the rejected EDRs through the pipeline again for another attempt to rate them.
8. The *pre-recycle pipeline* recycles or test recycles the suspended calls. The pre-recycle pipeline converts the suspended call objects back into files that the pipeline can process and routes the suspended call records back through their original pipeline for recycling.

If you are test recycling calls, the pipeline tries to rate the calls, but does not make any changes to the database.

3

Configuring Standard Recycling

Learn how to set up the Oracle Communications Billing and Revenue Management (BRM) standard recycling feature.

Topics in this document:

- About Configuring Standard Recycling
- Configuring Pipeline Modules for Standard Recycling
- Configuring a Preprocessing Pipeline
- Configuring Standard Recycling in a Rating Pipeline
- Configuring a Pre-Recycling Pipeline
- Configuring Recycle Request Handling
- Configuring a Pipeline Module to Add Recycle Keys to EDRs
- Configuring SE Loader for Standard Recycling

About Configuring Standard Recycling

Before configuring standard recycling, ensure you are familiar with Pipeline Manager and how to configure pipeline rating, EDR input process, EDR output process, and EDR preprocessing. For more information, see *BRM Setting Up Pipeline Pricing*.

Table 3-1 lists the tasks required for configuring standard recycling.

Table 3-1 Standard Recycling Configuration Tasks

Task	Description
1. Configure the pre-processing pipeline. See "Configuring a Preprocessing Pipeline".	• Configure your input module. • Configure the OUT_GenericStream pipeline module. • Configure MultiDB routing logic (optional).
2. Configure the rating pipeline. See "Configuring Standard Recycling in a Rating Pipeline".	• Configure the INP_GenericStream pipeline module. • Configure the FCT_PreSuspense pipeline module. • Configure the FCT_Reject pipeline module. • Set RejectStream to SuspenseCreateOutput. • Configure the FCT_Suspense or pipeline module. • Configure the **SuspenseCreateOutput** registry entry. • Configure the **SuspenseUpdateOutput** registry entry.

Table 3-1 (Cont.) Standard Recycling Configuration Tasks

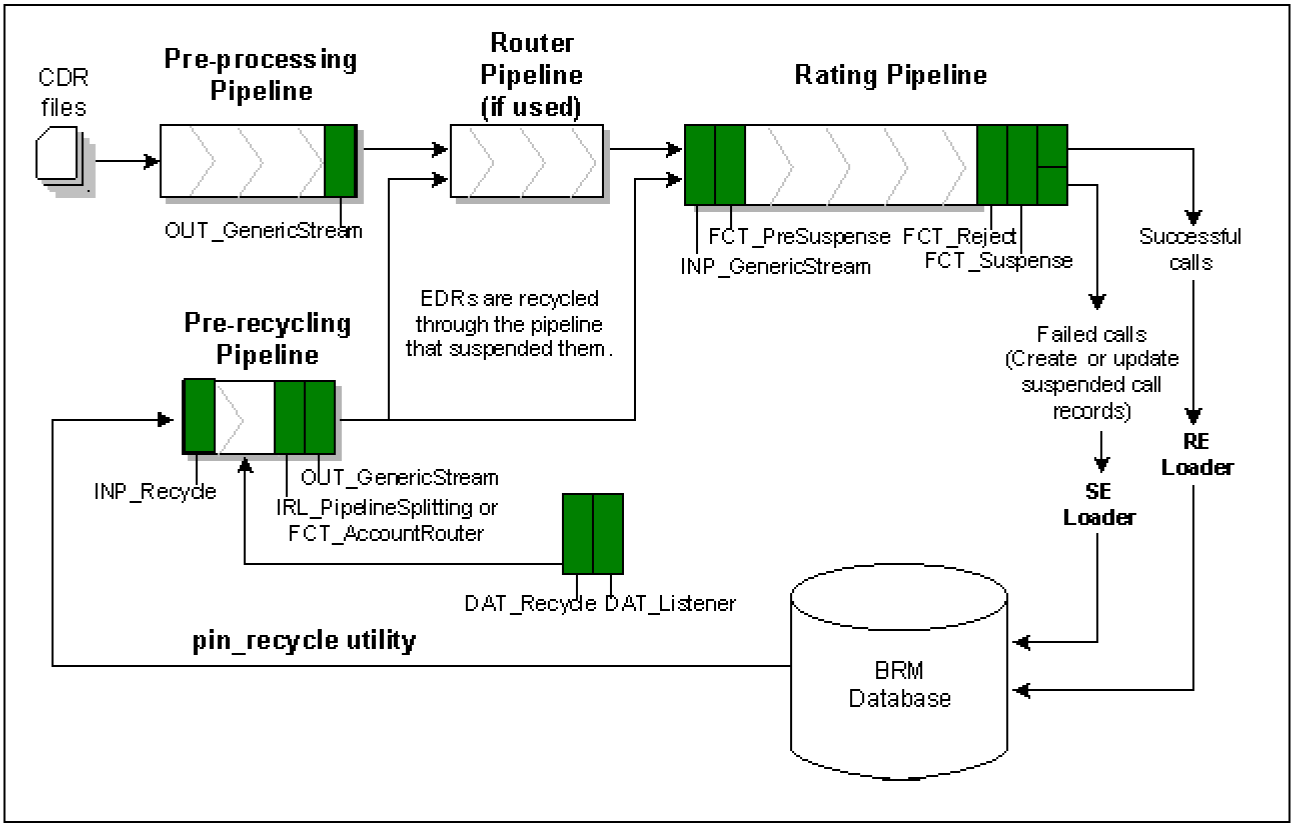
Task	Description
3. Configure the pre-recycling pipeline. See "Configuring a Pre-Recycling Pipeline".	• Configure the INP_Recycle pipeline module. • Edit the IRL_PipelineSplitting.data pipeline module. • Configure the OUT_GenericStream pipeline module.
4. Configure recycle request handling. See "Configuring Recycle Request Handling".	• Configure the DAT_Listener pipeline module. • Configure the DAT_Recycle pipeline module.
5. Configure a pipeline module to add recycle keys to EDRs (if needed). See "Configuring a Pipeline Module to Add Recycle Keys to EDRs".	• Used by features that temporarily suspend rating.
6. Set up Suspended Event (SE) Loader. See "Configuring SE Loader for Standard Recycling".	• Configure the Batch Controller. • Edit the **Infranet.properties** file.
7. Confirm that **pin_rel** is configured.	• Ensure that **pin_rel** is configured for standard recycling.

Configuring Pipeline Modules for Standard Recycling

Standard recycling requires you to configure the rating pipeline to handle suspended call records. For an example of a complete sample pipeline, see the *Pipeline_home***/conf/wireless.reg** file.

Figure 3-1 shows in green the pipeline modules that you need to configure.

Figure 3-1   Pipeline Modules to Configure for Standard Recycling



Configuring a Preprocessing Pipeline

Standard recycling requires a preprocessing pipeline, and this section explains how to set it up. For a complete example of a preprocessing pipeline, see *Pipeline_home***/conf/wireless.reg**.

All call records coming into your system for rating go through the preprocessing pipeline only once. After the preprocessing pipeline, EDRs go to the rating pipeline. Failed calls may be recycled through the rating pipeline multiple times, but they skip the preprocessing pipeline after going through once.

To configure your preprocessing pipeline:

1. Define preprocessing pipelines in the registry. You need a separate preprocessing pipeline for each input format your system uses.
2. Set up the input module of each pipeline to process call records from the external system you are using. You need a different pipeline for each call record format.
3. Configure the OUT_GenericStream pipeline module as an output module of the preprocessing pipeline.
   - Add this entry to the **DataDescription.StreamFormats** section of each preprocessing pipeline:

     ```
     SOL42=./FormatDesc/Solution42/SOL42_V670_REL.dsc
     ```

   - Add this entry to the **DataDescription.OutputMappings** section of each preprocessing pipeline:

```
SOL42=./FormatDesc/Solution42/SOL42_V670_REL_OutMap.dsc
```

- Add this entry to the output module section of each preprocessing pipeline:

  ```
  Grammar=./FormatDesc/Solution42/SOL42_V670_REL_OutGrammar.dsc
  ```

For complete examples of these registries, see *Pipeline_home***/conf/wireless.reg**. For details on this module, see "OUT_GenericStream" in *BRM Pipeline Manager Reference*.

4. (Optional) Multidatabase Manager users typically add a Multidatabase Manager routing pipeline after the preprocessing pipeline.

The preprocessing pipeline is now configured.

Configuring Standard Recycling in a Rating Pipeline

All calls go through a rating pipeline at least once, and suspended calls may be recycled through this pipeline multiple times. For a complete example of a rating pipeline, see *Pipeline_home***/conf/wireless.reg**.

> **Note:**
>
> You must use the input description file specified below. Customized description files are not supported.

To configure your rating pipeline:

1. Configure the INP_GenericStream pipeline module as the input module:
   - Add this entry to the **DataDescription.StreamFormats** section of each rating pipeline:

     ```
     SOL42_INPUT=./FormatDesc/Solution42/SOL42_V670_REL_ForInput.dsc
     ```

   - Add this entry to the **DataDescription.InputMapping**s section of each rating pipeline:

     ```
     SOL42_INPUT=./FormatDesc/Solution42/SOL42_V670_REL_InMap.dsc
     ```

   - Add this entry to the input module section of each rating pipeline:

     ```
     Grammar=./FormatDesc/Solution42/SOL42_V670_REL_InGrammar.dsc
     ```

   For examples of these entries, see *Pipeline_home***/conf/wireless.reg**. For details on this module, see "INP_GenericStream" in *BRM Pipeline Manager Reference*.
2. Configure FCT_PreSuspense as the first function module of the pipeline. For details, see "FCT_PreSuspense" in *BRM Pipeline Manager Reference*.
3. Configure FCT_Reject to route suspended calls to the suspense create output module (in Step 5). For details, see "How the FCT_Reject Module Works" and "FCT_Reject" in *BRM Pipeline Manager Reference*.
4. Set the **RejectStream** entry to **SuspenseCreateOutput** in the rating pipeline:

   ```
   ...
   ALL_RATE
   {
      Active                      = true

      CountryCode                 = 49
      MobileCountryCode           = 262
      NationalAccessCode          = 0
      InternationalAccessCode     = 00
   ```

```
   InternationalAccessCodeSign = +
   NetworkDestinationCode      = 172
   RejectStream                = SuspenseCreateOutput
...
```

5. Configure FCT_Suspense as the last function module of the rating pipeline. You need to configure the registry section of this module. For details, see "FCT_Suspense" in *BRM Pipeline Manager Reference*.

6. Confirm that your pipeline contains a **MaxErrorRates** output entry. If this entry is missing unexpected log file messages may result.

7. Configure the suspense create output module as one of the output modules for this pipeline.

   The following example works as a suspense create output module. Add the **/suspended_usage** object produced by this pipeline in the **EventType** entry:

   ```
   SuspenseCreateOutput
   {
      ModuleName                  = OUT_GenericStream
      EventType                   = /suspended_usage
      Module
      {
         Grammar                  = ./formatDesc/Formats/SuspenseHandling/
   SuspendedUsageCreationGrammr.dsc
         DeleteEmptyStream        = True
         OutputStream
         {
         ModuleName               = EXT_OutFileManager
         Module
         {
            OutputPath            = ./data/reject
            OutputPrefix          = suspense_create_
            OutputSuffix          = .out
            TempPrefix            = tmp

            TempDataPath          = ./data/reject
            TempDataPrefix        = susp.create.tmp.
            TempDataSuffix        = .data

            Replace               = True
            AppendSequenceNumber  = False
            }
         }
      }
   } # end of SuspenseCreateOutput
   ```

   > **Note:**
   >
   > To ensure output file integrity, specify a unique combination of **OutputPath**, **OutputSuffix**, and **OutputPrefix** values for each output stream defined in the registry.

8. Configure the suspense update output module as one of the output modules for this pipeline.

   This example implements a suspense output module:

   ```
   #-----------------------------
   SuspenseUpdateOutput
   ```

```
{
   ModuleName                  = OUT_GenericStream

   EventType                   = /tmp_suspended_usage

   Module
   {
      Grammar                  = ./formatDesc/Formats/SuspenseHandling/
SuspendedUsageUpdateGrammar.dsc
      DeleteEmptyStream        = True

      OutputStream
      {
         ModuleName            = EXT_OutFileManager
         Module
         {
         OutputPath            = ./data/reject
         OutputPrefix          = suspense_update_
         OutputSuffix          = .out
         TempPrefix            = tmp

         TempDataPath          = ./data/reject
         TempDataPrefix        = susp.update.tmp.
         TempDataSuffix        = .data

         Replace               = True
         AppendSequenceNumber  = False
         }
      }
   }
} # end of SuspenseUpdateOutput
```

> **Note:**
>
> To ensure output file integrity, specify a unique combination of OutputPath, OutputSuffix, and OutputPrefix values for each output stream defined in the registry.

Configuring a Pre-Recycling Pipeline

When suspended call records are recycled, they are first processed by a pre-recycling pipeline and then reprocessed by the original rating pipeline.

The pre-recycling pipeline used the INP_Recycle module. This module is used by standard recycling and Suspense Manager. It reads suspended usage records from the BRM database, restores original EDRs, applies edits to them, and pushes EDRs into the pre-recycling pipeline. For a complete example of a pre-recycling pipeline, see *Pipeline_home***/conf/wireless.reg**.

To configure your pre-recycling pipeline:

1. Configure INP_Recycle as the input module. For details, see "INP_Recycle" in *BRM Pipeline Manager Reference*.
   - In the EXT_InEasyDB module, change the **SqlDetail** entry to **StdRecycleDetail.sql**. See "EXT_InEasyDB" in *BRM Pipeline Manager Reference*.
2. Add and configure a pipeline module to send call records to the correct stream.

- **Single database schema systems:** Add and configure the IRL_PipelineSplitting module (an iRules module). Add this module to the pipeline registry iRules (its order in the registry is not important). Edit the **IRL_PipelineSplitting.data** file (in the *Pipeline_home*/**iScriptLib/iScriptLib_Suspense** directory), adding pipeline name/output stream pairs. For details, see "IRL_PipelineSplitting" and "FCT_IRules" in *BRM Pipeline Manager Reference*.
- **Multischema systems that require Account Migration Manager:** Add and configure the FCT_AccountRouter module. For details, see "FCT_AccountRouter" in *BRM Pipeline Manager Reference*.

> **Note:**
>
> AMM is not part of base BRM. Contact your BRM account manager for information about using AMM.

3. Configure the OUT_GenericStream pipeline module as an output module of the pre-recycling pipeline. Create a different OUT_GenericStream module for each rating pipeline used to recycle suspended calls.

   - Add this entry to the **DataDescription.StreamFormats** section of the pre-recycling pipeline:

     ```
     SOL42=./FormatDesc/Solution42/SOL42_V670_REL.dsc
     ```

   - Add this entry to the **DataDescription.OutputMappings** section of the pre-recycling pipeline:

     ```
     SOL42=./FormatDesc/Solution42/SOL42_V670_REL_OutMap.dsc
     ```

   - Add this entry to each output module section of the pre-recycling pipeline:

     ```
     Grammar=./FormatDesc/Solution42/SOL42_V670_REL_OutGrammar.dsc
     ```

   For details, see "OUT_GenericStream" in *BRM Pipeline Manager Reference*.

   Your pipelines are now ready to accept call records.

Configuring Recycle Request Handling

To configure recycle request handling, add DAT_Recycle to the registry data pool. For example:

```
RecyclingData
{
   ModuleName = DAT_Recycle
   Module
   {
      Listener      = ifw.DataPool.Listener
      LogEvents     = True
      ControlPath   = ./database/Oracle/Scripts/Suspense
      ParameterFile = parameter.isc
   }
}
```

For details, see "DAT_Recycle" in *BRM Pipeline Manager Reference*.

Configuring a Pipeline Module to Add Recycle Keys to EDRs

Programs and features that must temporarily interrupt and then restart rating, such as account migration, use the **pin_recycle** utility to recycle all EDRs after the interruption is over. The BRM features that add recycle keys to EDRs all have pipeline modules for doing this. See the feature documentation for details.

Configuring SE Loader for Standard Recycling

The procedures for installing, configuring, and using SE Loader are identical to those of RE Loader, except for the step listed here. For details, see *BRM Loading Rated Events*.

To configure SE Loader for standard recycling, do the following:

1. Add a separate instance of SE Loader to each pipeline.
2. Create a new *BRM_home*/**apps/pin_rel/suspense** directory by copying the contents of *BRM_home*/**apps/pin_rel/gsm/tel** to *BRM_home*/**apps/pin_rel/suspense**.
3. Confirm that these files are in the *BRM_home*/**apps/pin_rel/suspense** directory:
   - **pin.conf**
   - **SampleRelHandler_config.values**
   - **SampleRelHandler.pl**
4. Add these entries to the *BRM_home*/**apps/pin_rel/suspense/SampleRelHandler_config.values** file:

   ```
   $FILETYPE    = "*.out.bc";
   $HANDLER_DIR = "BRM_home/apps/pin_rel/suspense";#
   ```

5. Edit the *BRM_home*/**apps/batch_controller/Infranet.properties** file, adding **SUSPENSE** and **RECYCLE_ROLLBACK** entries to **batch.random.events**:

   ```
   batch.random.events     TEL,SMS,FAX,DATA,GPRS,SUSPENSE,RECYCLE_ROLLBACK
   ```

   Add these parameters to the new entries:

   ```
   #for SUSPENSE events:
   SUSPENSE.name                             SUSPENSE Usage
   SUSPENSE.handlers                         suspHandler
   SUSPENSE.file.location                    Pipeline_home/data/reject
   SUSPENSE.file.pattern                     suspense_ *.out

   suspHandler.name                          suspHandler
   suspHandler.max.at.highload.time          1
   suspHandler.max.at.lowload.time           1
   suspHandler.start.string                  BRM_home/apps/pin_rel/suspense /
   SampleRelHandler.pl

   #For RECYCLE_ROLLBACK events:
   RECYCLE_ROLLBACK.name                     RECYCLE_ROLLBACK Usage
   RECYCLE_ROLLBACK.handlers                 recycleRollbackHandler
   RECYCLE_ROLLBACK.file.location            Pipeline_home/data/error
   RECYCLE_ROLLBACK.file.pattern             testDB*.err

   recycleRollbackHandler.name               recycleRollbackHandler
   recycleRollbackHandler.max.at.highload.time 1
   recycleRollbackHandler.max.at.lowload.time  1
   ```

```
recycleRollbackHandler.start.string          BRM_home/apps/pin_rel/recycle/
SampleRelHandler.pl
```

6. Confirm that these *BRM_home*/**apps/pin_rel/Infranet.properties** file entries are set to **false**:

   ```
   infranet.rel.validate_dbnumber = false
   infranet.rel.validate_indexes  = false
   ```

   > **Note:**
   >
   > The SE Loader architecture makes obsolete the database consistency checks and number validation controlled by these entries.

7. Create a new *BRM_home*/**apps/pin_rel/recycle** directory by copying the contents of *BRM_home*/**apps/pin_rel/gsm/tel** to *BRM_home*/**apps/pin_rel/recycle**.

8. Add these entries to the *BRM_home*/**apps/pin_rel/recycle/SampleRelHandler_config.values** file:

   ```
   $FILETYPE    = "*.err.bc";
   $HANDLER_DIR = "BRM_home/apps/pin_rel/recycle";#
   ```

9. Add this entry to each output stream of the pre-recycling pipeline in your *Pipeline_home*/**conf/wireless.reg** file:

   ```
   EventType = /recycle_suspended_usage
   ```

4

Using Standard Recycling to Recycle Suspended EDRs

You use the Oracle Communications Billing and Revenue Management (BRM) **pin_recycle** utility, part of the standard recycling feature, to recycle suspended EDRs.

Topics in this document:

- About Recycling Using pin_recycle
- About the Standard Recycling Mechanism
- Setting Up EDR Recycling by CDR File
- About Recycling Suspended EDRs After Rating Interruptions
- Setting Up EDR Recycling by Recycle Key
- Setting Up pin_recycle to Run Periodically

About Recycling Using pin_recycle

EDRs are usually recycled for one of two reasons:

- The pipeline suspended them because of a problem with the pipeline or EDR. After fixing the problem, you recycle the EDRs by using the BRM standard recycling tools in another attempt to rate them. The standard recycling tools recycle all EDRs from the same CDR file at the same time.
- A BRM program suspended them intentionally due to a temporary interruption in rating. These programs mark the EDRs with a recycle key and store them until the interruption ends. All EDRs with the same recycle key are recycled at the same time.

In both cases, you use **pin_recycle** to recycle the suspended EDRs back through the pipeline, rate them, and capture the revenue they represent.

For information about configuring Pipeline Manager to suspend calls, see "Installing and Configuring Suspense Manager".

About the Standard Recycling Mechanism

The BRM standard recycling mechanism uses the FCT_Reject, FCT_Suspense, and FCT_PreSuspense pipeline modules, along with the **pin_recycle** utility, to suspend and recycle calls from the same CDR input file. After examining the Pipeline Manager log files to determine why calls were suspended, Pipeline Manager administrators fix the pipeline and then use the utility to attempt to rate these calls again. For details about setting up and using the standard recycling tools, see "Configuring Standard Recycling".

Configuring the pipeline requires system administration experience. You must be familiar with:

- Modifying BRM pipeline modules to append EDRs with data. For details on setting up and administering pipeline rating, see "About Pipeline Rating" in *BRM Setting Up Pipeline Pricing*.

- Creating a **crontab** file entry to run the **pin_recycle** utility to recycle or delete EDRs. See your operating system documentation for details on creating a **cron** command.

Setting Up EDR Recycling by CDR File

To set up BRM to recycle EDRs by CDR file:

1. Configure the pipeline to reject EDRs according to your business policies. For details, see "Configuring Standard Recycling".
2. Run the **pin_recycle** utility with the **-f** option to recycle suspended records within a CDR file. You can test recycle, recycle, or delete all failed EDRs in that CDR file. For the complete **pin_recycle** syntax, see "pin_recycle" in *BRM Pipeline Manager Reference*.

   You can run this utility like any other BRM utility, but you will probably want to run it manually as needed. How often you run this script depends on how many EDRs your pipeline rejects. When you make frequent or significant changes to your pipeline, you need to check your log files frequently. If many EDRs are being rejected, run **pin_recycle** often.

About Recycling Suspended EDRs After Rating Interruptions

Some BRM programs and features temporarily interrupt and then restart rating for specific accounts. These programs and features use **pin_recycle** to recycle calls for those accounts when the interruption is over. These features, such as account migration and pipeline-triggered billing, temporarily stop rating by directing the pipeline to suspend calls that come in during the interruption. These call records arrive in the pipeline and are appended with a recycle key. When the interruption is over, you use **pin_recycle** to rate all the stored calls that contain that recycle key. You can further configure this feature by using any number of different recycle keys to control when suspended EDRs get recycled.

> **Note:**
>
> This feature is compatible with both Suspense Manager and standard recycling.

The **-k** *recycle key* option directs **pin_recycle** to search for all EDRs that contain a specific recycle key string and a **suspended** status and queue them for rating. The BRM feature that suspends EDRs determines which EDRs contain the same recycle key and need to be recycled together. This gives **pin_recycle** the flexibility to restrict recycling to just the EDRs with specific characteristics selectively.

For example, the account migration feature moves groups of accounts across databases and temporarily stops rating each group of accounts while they are being moved. Account migration uses internal job IDs to keep track of the accounts being moved. It also uses these job IDs in the recycle keys for suspended EDRs associated with those same accounts.

In contrast, the pipeline-triggered billing feature interrupts all rating for all accounts. Therefore, pipeline-triggered billing only needs one recycle key (**Trigger_Billing**) for all EDRs that arrive during the temporary suspension.

Before using **pin_recycle**, you must configure a pipeline module to add the recycle key. For details, see "Setting Up EDR Recycling by Recycle Key".

Setting Up EDR Recycling by Recycle Key

To set up BRM to recycle suspended EDRs by the recycle key:

1. Configure the pipeline to suspend EDRs according to your business policies. For details, see "Configuring Standard Recycling".
2. Configure BRM to add the recycle key to EDRs during temporary interruptions. The feature requiring the temporary interruption has a pipeline module associated with it that does this. For example, the pipeline-triggered billing feature uses the FCT_TriggerBilling module, and Account Migration Manager uses the FCT_AccountRouter module.

   > **Note:**
   >
   > AMM is not part of base BRM. Contact your BRM account manager for information about using AMM.

   The recycle key can be any string that corresponds to a set of EDRs to recycle. You configure a pipeline module to add your recycle key to the DETAIL.ASS_SUSPENSE_EXT.RECYCLE_KEY field of each EDR. The specific module depends on the program running billing and your recycling strategy.
3. Configure **pin_recycle** to run periodically. You do this by adding it to a **cron** job. For details, see "Setting Up pin_recycle to Run Periodically".
4. Configure **pin_recycle** to run periodically with the **-d** option to remove successfully recycled EDRs from the BRM database. You can do this by adding **pin_recycle** to a **cron** job.

Setting Up pin_recycle to Run Periodically

Run **pin_recycle** periodically to queue the temporarily stored EDRs for rating and to delete them. The **cron** command is the typical way to do this, although you can run **pin_recycle** like any other BRM command-line utility. This section explains how to set up **cron** command to run **pin_recycle**.

You need to add two **pin_recycle** entries to the **cron** command. One is to search for and recycle EDRs, and the other is to delete them after they are recycled.

Adding EDR Recycle Entries

To run **pin_recycle** periodically, add entries like the following. The optimal frequency depends on your recycling strategy.

Use a **cron** job with a **crontab** entry to run the **pin_recycle** script. The following **crontab** entry runs **pin_recycle** at 1:00 a.m. daily, and queues EDRs with a recycle key of **Trigger_Billing** for rating:

```
0 1 * * * BRM_home/bin/pin_recycle -k Trigger_Billing &
```

BRM_home is the directory where you installed BRM components.

Adding EDR Delete Entries

To remove EDRs from the BRM database, add an entry like the following. The optimal frequency depends on your recycling strategy.

Use a **cron** job with a **crontab** entry to run the **pin_recycle** script. The following **crontab** entry runs **pin_recycle** at 1:00 a.m. daily, and deletes EDRs with a recycle key of **Trigger_Billing**:

```
0 1 * * * BRM_home/bin/pin_recycle -k Trigger_Billing -d &
```

5
About Suspense Manager

Learn about the features in Oracle Communications Billing and Revenue Management (BRM) Suspense Manager.

Topics in this document:

- About Suspense Manager
- About Upgrading from Standard Recycling to Suspense Manager

> **Note:**
>
> - Suspense Manager is an optional component, not part of base BRM.
> - Suspense Manager is an extension of the BRM standard recycling feature. You must configure standard recycling before configuring Suspense Manager.

About Suspense Manager

You use Suspense Manager to:

- Analyze, edit, recycle, write off, archive, restore, and delete individual CDRs that have failed pipeline processing.
- Any number of individual call records at the same time (bulk processing).
- Analyze, resubmit, write off, and delete CDR files containing any number of individual call records. CDR files cannot be edited or archived.

> **Note:**
>
> Suspense Management Center and certain BRM utilities and tools refers to CDR files as *batches* or *batch files*.

Suspense Manager includes Suspense Management Center, which allows you to perform these tasks using a graphical user interface.

Records "fail" if Pipeline Manager cannot process them. These failures can be intentional or inadvertent. For example:

- A call record may arrive with invalid data and fail a pipeline validity rule.
- The Pipeline Manager database tables are set up incorrectly.
- The call record may fail custom validity checking that you have set up in a custom iScript, such as a size or time duration limit for individual records.

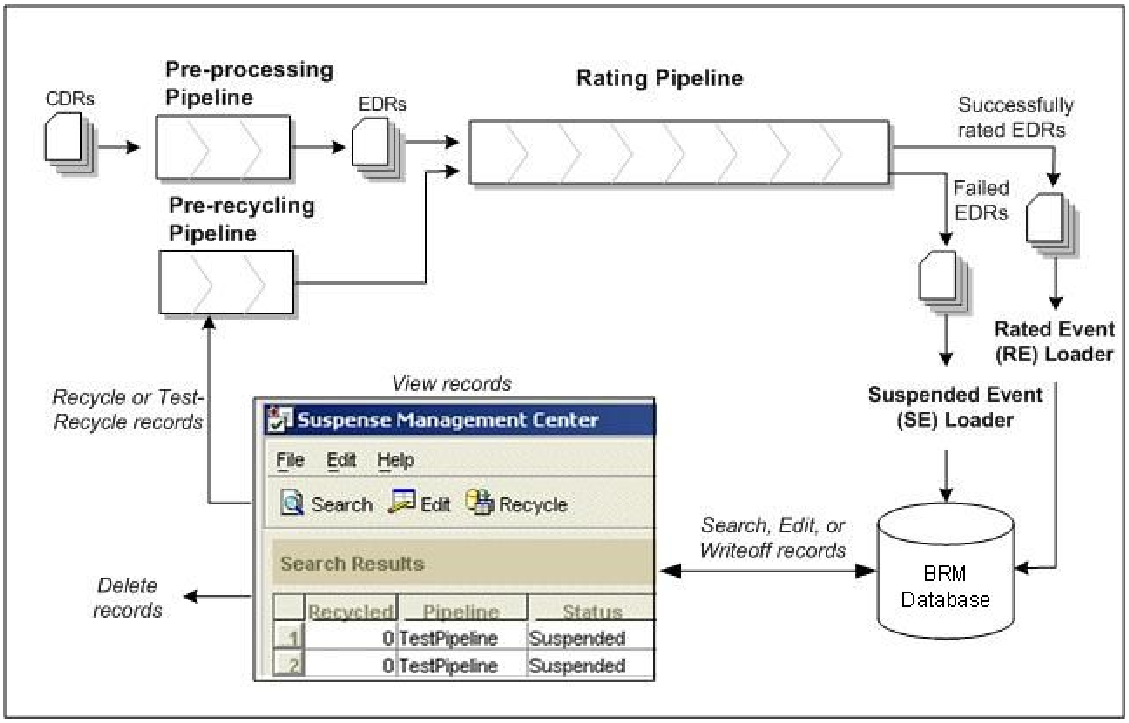
Suspense Manager replaces or augments the base BRM standard recycling feature for rejecting or recycling suspended calls.

Suspense Manager server components are available on Linux operating system and require Oracle database software. The Suspense Management Center client application runs on Windows systems. For details on system requirements, see *BRM Compatibility Matrix*.

Suspending Individual CDRs, or CDRs in Bulk

Figure 5-1 shows an example of how Suspense Manager can fit into your BRM system to manipulate individual failed CDRs, on groups of CDRs at once.

Figure 5-1   Suspense Manager and Individual Failed CDRs in BRM



	Recycled	Pipeline	Status
1	0	TestPipeline	Suspended
2	0	TestPipeline	Suspended

In the example above, CDRs first enter Suspense Manager through a *preprocessing pipeline*. The preprocessing pipeline converts these records to the format used by Suspense Manager. These records go through the preprocessing pipeline only once, and only a few modules are needed for it.

This example shows EDRs next going through a normal *rating pipeline*. Most pipeline function modules are included here. CDR "success" and "failure" policies are configured in this pipeline. If records "fail" in this pipeline, they are directed to the appropriate event loader to be loaded into the database.

The *SE Loader* converts the failed calls to objects in the BRM database.

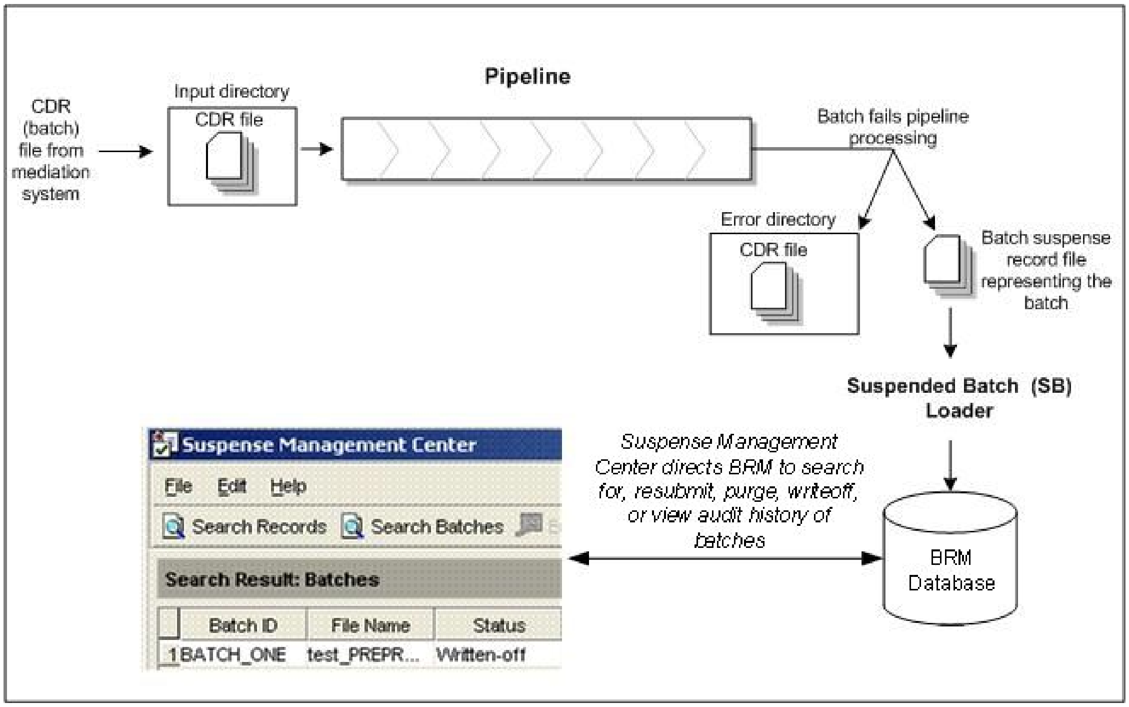
Then, you manipulate these records by using the *Suspense Management Center* application. This application allows you to search for, edit, undo edits, test recycle, recycle, write off, or delete suspended CDRs or CDR files.

In this example, suspended records that get recycled are processed by the *pre-recycle pipeline* before they go through the rating pipeline again. Before recycling these records, you would probably make changes to the rating pipeline or edit them so they are successfully rated when they go through the pipeline again. The pre-recycle pipeline converts the suspended record objects back into files that the pipeline can process, and routes the suspended records back through their original pipeline for recycling.

Suspending CDR Files

Figure 5-2 shows an example of how Suspense Manager can fit into your BRM system to manipulate files containing multiple CDRs.

Figure 5-2   Suspense Manager and Files Containing Multiple CDRs


Pipeline
CDR (batch) file from mediation system
Input directory
CDR file
Batch fails pipeline processing
Error directory
CDR file
Batch suspense record file representing the batch
Suspended Batch (SB) Loader
Suspense Management Center
File Edit Help
Search Records Search Batches
Search Result: Batches
Batch ID File Name Status
1 BATCH_ONE test_PREPR... Written-off
Suspense Management Center directs BRM to search for, resubmit, purge, writeoff, or view audit history of batches
BRM Database

This example shows a CDR file entering through a mediation system. The CDR file is first placed in the pipeline's input directory and then processed by the pipeline. The pipeline contains "success" and "failure" policies based on file-level or record-level validation. If the CDR file fails pipeline processing, it is directed to the pipeline's error directory, and a **batch_suspense_create** file is created. SB Loader uses the information in this file to create an object, which is stored in the BRM database.

You use the Suspense Management Center GUI to manipulate suspended CDR files by acting on the suspended CDR file objects. Using Suspense Management Center, you can resubmit CDR files through the pipeline, purge them, write them off, or view their audit histories.

If the problem with the CDR file is a bad pipeline policy or configuration, you can correct the pipeline and resubmit the CDR file for another attempt at processing.

You have several options if the problem is with the CDR file itself. You can force the pipeline to ignore certain errors and process the CDR file, or you can give up on the CDR file and purge it from the database. The list of pipeline errors to ignore is configurable and must be set up beforehand. You cannot edit CDR files using Suspense Management Center.

Suspended Call Record States

As Suspense Manager processes suspended records, they are assigned one of the following states:

- **Suspended**: Pipeline Manager could not process the call record, so it has been stored in the BRM database as a suspended call record.
- **Recycling**: The call record is being sent through the rating pipeline again to be rated.
- **Succeeded**: The call record has been successfully recycled and rated.
- **Written off**: The call will not be recycled but will be stored for further use.

Table 5-1 lists the details about the states.

Table 5-1 Suspended Call Record States

State	PIN_FLD_STATUS value in /suspended_usage	Can be edited	Can be recycled	Can be written off	Can be deleted	Can be archived and deleted
Suspended	**0**	Yes	Yes	Yes	No	No
Recycling	**1**	No	No	No	No	No
Succeeded (Successfully processed)	**2**	No	No	No	Yes	Yes
Written off	**3**	No	No	No	Yes	Yes

About SE and SB Loaders

The Suspended Event (SE) and Suspended Batch (SB) loaders load suspended records into the BRM database but operate on different types of records. The SE loader takes suspended (failed) CDRs as input and uses the **pin_rel** utility to load them into the BRM database as **/suspended_usage** objects. This utility is usually set up to run automatically, but you can run it manually as needed.

The SE loader is a special Rated Event Loader (RE Loader) configuration, which loads prerated wireless events into the BRM database as objects.

The SB loader does not load CDR files directly into the BRM database. Instead, it accepts information from the **suspense_create_batch** file created for each failed CDR file and creates **/suspended_batch** objects. The SB loader uses the **load_suspended_batch_info.pl** script to create the **/suspended_batch** objects. This script is usually set up to run automatically, but you can run it manually as needed.

About the FCT_BatchSuspense Module

The FCT_BatchSuspense module adds suspense reason and subreason codes to batches.

- If a resubmitted batch is successful, FCT_BatchSuspense generates a **batch_suspense_update** file with a **Succeeded** status. The SB loader reads this file and updates the corresponding **/suspended_batch** object to **Succeeded** when you run **load_suspended_batch_info.pl**.
- If a resubmitted batch fails again, FCT_BatchSuspense generates a **batch_suspense_update** file with a **Suspended** status,a new error code, and a new

suspense reason. The SB loader reads this file and updates the corresponding **/suspended_batch** object with a **Suspended** status.

The specific errors that the FCT_BatchSuspense module adds are based on the error codes assigned to the EDR by the pipeline and the mapping information stored in the **/config/batch_suspense_reason_code** object. If no **/config/batch_suspense_reason_code** object is present, this module sets the suspense reason to **O** (other).

See "FCT_BatchSuspense" in *BRM Pipeline Manager Reference*.

Differences Between the RE, SE, and SB Loaders

Table 5-2 explains the differences between the three event loaders.

Table 5-2   Differences Between Event Loaders

Task	RE Loader	SE Loader	SB Loader
Loads these types of records	Event file	CDRs	CDR files
Creates these objects	**/event**	**/suspended_usage**	**/suspended_batch**

Suspense Manager Objects

Suspense Manager stores individual suspended CDRs in **/suspended_usage** objects and suspended CDR file records in **/suspended_batch** objects. During configuration, you create a subclass of these objects for each type of call record you receive.

Every action performed by Suspense Management Center is recorded in these **/admin_action** objects:

- **/admin_action/suspended_usage/edit**
- **/admin_action/suspended_usage/recycle**
- **/admin_action/suspended_usage/writeoff**
- **/admin_action/suspended_batch**
- **/admin_action/suspended_batch/resubmit**
- **/admin_action/suspended_batch/writeoff**

For example, when you edit multiple suspended call records at the same time, Suspense Manager records the edits in an **/admin_action/suspended_usage/edit** object. All of the individual suspended CDRs have **/suspended_usage/***type* objects that reference the **/admin_action/suspended_usage/edit** object.

If you choose to override specific suspense reasons during recycling, the reasons available to override are stored in **/config/suspense_override_reason** objects.

About Upgrading from Standard Recycling to Suspense Manager

To upgrade a system from standard recycling to the Suspense Manager features, follow the instructions in "Installing and Configuring Suspense Manager".

> **Note:**
>
> Suspense Manager is an optional component, not a part of base BRM.

For details about using Suspense Management Center with call records created using standard recycling, see "Using Suspense Management Center with Standard Recycling Call Records".

6

Installing and Configuring Suspense Manager

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

Topics in this document:

- Installing Suspense Manager
- About Configuring Suspense Manager
- Planning and Setting up Your Database for Suspense Manager
- Creating a List of Editable Fields Based on Your /suspended_usage Subclasses
- Loading Editable Fields into the Database
- Changing the List of Suspense Reasons and Subreasons
- Configuring Pipeline Manager for Suspense Manager
- Setting Up Suspended Event (SE) Loader for Suspense Manager
- Setting Up Suspended Batch (SB) Loader for Suspense Manager
- Creating Indexes for Search Templates
- Configuring and Customizing Suspense Management Center
- Configuring Event Notification for Suspense Manager
- Configuring Debugging (Optional)
- Configuring the Number of Suspended Records to Process in a Transaction
- Suspense Management Center Permission Types

Installing Suspense Manager

Before installing Suspense Manager, you must install BRM and Rated Event (RE) Loader, which is an optional BRM component. For details, see "Installing RE Loader" in *BRM Loading Rated Events*.

To install and set up Suspense Manager, install the following components:

- Suspense Manager. See "Installing Individual BRM Components" in *BRM Installation Guide*.
- Suspense Management Center client software. See "Installing Individual BRM Clients" in *BRM Installation Guide*.

About Configuring Suspense Manager

The business decisions you make in the planning phase determine the details of your implementation in the configuration phase.

Table 6-1 shows the tasks required for configuring Suspense Manager.

Table 6-1 Tasks to Configure Suspense Manager

Task	Description
1. Plan and set up your database. See "Planning and Setting up Your Database for Suspense Manager".	• Decide whether to extend the **/suspended_usage** class. • Select a list of EDR fields that can be queried. • Add **/suspended_usage** subclasses with fields that can be queried.
2. Create a list of editable fields. See "Creating a List of Editable Fields Based on Your /suspended_usage Subclasses".	• Create a list of fields that you can correct using Suspense Manager.
3. Load editable fields into the database. See "Loading Editable Fields into the Database".	• Edit the **pin_suspense_editable_flds** file. • Run the **load_pin_suspense_editable_flds** utility.
4. Change suspense reasons and subreasons (optional). See "Changing the List of Suspense Reasons and Subreasons".	• Decide whether to change the suspense reason lists. • Edit the **suspense_reason_code.en_US** file or the **batch_suspense_reason_code.en_US** file. • Edit the **pin_suspense_reason_code** or **pin_batch_suspense_reason_code** file. • Run the **load_localized_strings** utility. • Run the **load_pin_suspense_reason_code** or **load_pin_batch_suspense_reason_code** utility.
5. Configure the pipeline for Suspense Manager. See "Configuring Pipeline Manager for Suspense Manager".	• Configure the standard recycling pipeline. • Configure the rating pipeline. • Configure the pre-cycling pipeline.
6. Configure SE or SB Loader. See "Setting Up Suspended Event (SE) Loader for Suspense Manager" or "Setting Up Suspended Batch (SB) Loader for Suspense Manager".	• Edit the **Infranet.properties** file.
7. Create indexes for search templates. See "Creating Indexes for Search Templates".	• Create indexes.
8. Configure and customize Suspense Management Center. See "Configuring and Customizing Suspense Management Center".	• Add custom fields (edit the **custom.properties** file). • Add custom fields to Web Start (edit **SuspenseManagement_en.jnlp**) (Optional). • Set up permissions for Suspense Management Center.
9. Configuring event notification for Suspense Manager. See "Configuring Event Notification for Suspense Manager".	• Consolidate event notification operations.
10. Configure debugging (optional). See "Configuring Debugging (Optional)".	• Set up Java PCM logging (edit the **Infranet.properties** file). • Edit **RunSM.bat**.
11. Configure the number of records to process in a transaction (optional). See "Configuring the Number of Suspended Records to Process in a Transaction".	• Edit the **pin_suspense_params** file and run **load_suspense_params**.

Planning and Setting up Your Database for Suspense Manager

> **Note:**
>
> The planning process is critical to the successful operation of Suspense Manager. During setup, you change the database schema. Changing the database schema *after* you start using Suspense Manager requires a database upgrade that can be time-consuming and painful. Be sure to follow the steps in this section carefully.

The process of setting up your database involves these two steps:

- Picking the EDR fields you will use to search for suspended EDRs (queryable fields). If the queryable fields your implementation requires are not in the default BRM objects, you need to create new object subclasses for them.
- Picking a list of fields that you will allow your personnel to edit. BRM assumes this list is a subset of the default EDR fields and any new queryable fields. If not, then you will need to create new objects for them.

After you decide which fields to add or edit in EDRs, you will load the list into the BRM database, which will allow Suspense Manager to access them.

Deciding Whether You Need to Extend the Suspense Subclasses

Your business will require searching for and analyzing suspended call records (individual CDRs or CDR files). The first step in setting up Suspense Manager is to decide whether the default Suspense Manager storable classes meet your needs. The storable classes you use must contain the fields your business requires to search for the records you need and analyze their problems. If not, you will need to modify, extend, or replace the default storable classes.

Selecting a List of Queryable EDR Fields

You will use this list of EDR fields to search for and analyze call records. Suspense Management Center allows you to search for suspended calls based on values in these queryable fields and displays these values in your search results.

> **Note:**
>
> After you specify and load the list of queryable fields into the database, modifying the list involves significant effort. Be sure to plan the list carefully.

Each BRM implementation requires a different list. Making all of your EDR fields queryable degrades performance by using a lot of unnecessary database storage. However, make enough fields queryable to meet your business needs.

Start by reviewing the object specifications for the sample Suspense Manager storable classes listed below. If these storable classes contain all the information your business requires, you won't need to make any changes and can skip the next section. If you need to expand or extend these classes, list the fields you want to make queryable. Use this list to define custom extensions to **/suspended_usage** object types.

> **Note:**
>
> Add one set of queryable fields representing one **/suspended_usage** subclass *per pipeline*. For example, for a single pipeline that accepts **/suspended_usage/telco/gsm** records, you can select queryable fields from the **/suspended_usage/telco** and **/suspended_usage/telco/gsm** subclasses. You cannot select queryable fields from **/suspened_usage/telco/gprs**, because it requires a separate pipeline.

/suspended_usage/telco

This storable class stores general wireless call record data in the fields listed in Table 6-2.

Table 6-2 Fields in /suspended_usage/telco

Field	Description
PIN_FLD_BYTES_IN	Volume of data sent.
PIN_FLD_BYTES_OUT	Volume of data received.
PIN_FLD_CALLED_TO	Phone number being called.
PIN_FLD_CALLING_FROM	Phone number the call originated from.
PIN_FLD_CALL_DURATION	Call time duration.
PIN FLD PRIMARY MSID	Primary MSID.
PIN FLD SERVICE TYPE	Basic service type.
PIN_FLD_START_TIME	Time the call was initiated. **Note**: The start time is not stored in UTC format.
PIN_FLD_USAGE_TYPE	Describes the call scenario, for example, customer-to-customer call, birthday call, closed-user-group call, or friends & family call.

/suspended_usage/telco/gprs

This class stores call record data for generic GPRS (data) calls in the fields listed in Table 6-3.

Table 6-3 Fields in /suspended_usage/telco/gprs

Field	Description
PIN_FLD_BYTES_IN	Volume of data sent.
PIN_FLD_BYTES_OUT	Volume of data received.
PIN_FLD_CALLED_TO	Phone number being called.
PIN_FLD_CALLING_FROM	Phone number the call originated from.
PIN_FLD_CALL_DURATION	Call time duration.
PIN FLD PRIMARY MSID	Primary MSID.
PIN FLD SERVICE TYPE	Basic service type.

Table 6-3 (Cont.) Fields in /suspended_usage/telco/gprs

Field	Description
PIN_FLD_START_TIME	Time the call was initiated. **Note**: The start time is not stored in UTC format.
PIN_FLD_USAGE_TYPE	Describes the call scenario, for example, customer-to-customer call, birthday call, closed-user-group call, or friends & family call.
PN_FLD_APN	APN address.
PIN_FLD_GGSN_ADDRESS	GGSN address.
PIN_FLD_NODE_ID	Node ID.
PIN_FLD_SECONDARY_MSID	MSI number.
PIN_FLD_SGSN_ADDRESS	SGSN address.

/suspended_usage/telco/gsm

This class stores call record data for generic GSM (voice) calls in the fields listed in Table 6-4.

Table 6-4 Fields in /suspended_usage/telco/gsm

Field	Description
PIN_FLD_BYTES_IN	Volume of data sent.
PIN_FLD_BYTES_OUT	Volume of data received.
PIN_FLD_CALLED_TO	Phone number being called.
PIN_FLD_CALLING_FROM	Phone number the call originated from.
PIN_FLD_CALL_DURATION	Call time duration.
PIN FLD PRIMARY MSID	Primary MSID.
PIN FLD SERVICE TYPE	Basic service type.
PIN_FLD_START_TIME	Time the call was initiated. **Note**: The start time is not stored in UTC format.
PIN_FLD_USAGE_TYPE	Describes the call scenario, for example, customer-to-customer call, birthday call, closed-user-group call, or friends & family call.
PIN_FLD_CELL_ID	Network cell ID where the call originated.
PIN_FLD_DESTINATION_SID	Destination MSC or switch ID.
PIN_FLD_DIALED_NUMBER	Number that was called.
PIN_FLD_ORIGIN_SID	Origin MSC or switch ID.
PIN_FLD_SECONDARY_MSID	IMSI field.

All of the fields in subclasses of **/suspended_usage** are queryable.

Adding /suspended_usage Subclasses with Queryable Fields

Suspense Manager uses the **/suspended_usage** storable class to store failed call records in the BRM database. You must extend this class for each type of suspended call record your business requires. Suspense Manager provides the **/suspended_usage/telco**, **/**

suspended_usage/telco/gsm, and **/suspended_usage/telco/gprs** default subclasses to store data for suspended calls.

To add new types of call records to Suspense Manager:

1. Determine how to subclass **/suspended_usage** objects.
2. Create custom **/suspended_usage** objects.

   Use the Storable Class Editor to add a subclass to **/suspended_usage** for each type of call record your business uses.

Creating a List of Editable Fields Based on Your /suspended_usage Subclasses

You use Suspense Management Center to correct these fields in failed EDRs, recycle the calls, and rate them.

Review these objects and create the fields you need to change to correct a failed call. All fields you added to **/suspended_usage** subclasses are eligible for editing. You can change this list anytime by following the steps in "Loading Editable Fields into the Database".

Loading Editable Fields into the Database

To load your list of editable fields into the database for use by Suspense Management Center:

1. Review the list of fields you assembled in "Creating a List of Editable Fields Based on Your /suspended_usage Subclasses".
2. Add these fields to the *BRM_home***/sys/data/config/pin_suspense_editable_flds** file.
3. Run the **load_pin_suspense_editable_flds** utility (located in *BRM_home***/bin**) to load the editable fields into the database:

   ```
   load_pin_suspense_editable_flds  pin_suspense_editable_flds
   ```

   For details, see "load_pin_suspense_editable_flds" in *BRM Suspending and Recycling Event Records*.

Changing the List of Suspense Reasons and Subreasons

Suspense Manager adds the specific reasons for call failures to the EDRs it stores. These reasons, called *suspense reasons,* can be divided into more specific *suspense subreasons*. These suspense reasons and subreasons are stored in the call record along with the rest of the call record data. Because they are stored in the call records, you can search for them by using Suspense Management Center. For example, you can search for all the calls that could not be associated with a subscriber.

The Pipeline Manager error messages that actually cause call failures are mapped to these suspense reasons and subreasons. An extensive default error code-to-suspense reason mapping is provided in Suspense Manager. If your business requires different suspense reasons or subreasons, you can change them and their mappings. You can make these changes at any time, but because you may have to upgrade existing data, having this mapping in place is best before you go into production.

Deciding Whether to Change the Suspense Reason and Subreason Lists

Each suspense reason covers a group of Pipeline Manager error codes. The strings that define these error messages are listed in the *BRM_home***/sys/msgs/suspense_reason_code/suspense_reason_code.en_US** file. These error code strings are mapped to Pipeline Manager error codes in the *BRM_home***/sys/data/config/pin_suspense_reason_code** file.

If you need to change the default mapping or add additional reasons or subreasons, continue with the instructions in "Changing the List of Suspense Reasons and Subreasons" that describe the process.

Changing the Suspense Reason and Subreason Lists

If the default error messages or mappings do not meet your business needs, follow the steps in this section to change them. First, edit the text file with your new suspense reasons and subreasons, and then load the mapping into the database.

To change the default suspense reasons:

1. Determine your new suspense reasons.

   This list of the most common problems that cause calls to fail. It can be as extensive as you like. You can also look at these as *categories* of suspense reasons because this is the first of two levels. For example, on one level, you might use "Validation check failed" and then add more specific subreasons on the second level, such as "Call exceeds maximum time" or "Suspiciously long call".

   > **Note:**
   >
   > Any Pipeline Manager error message without a suspense reason will use the default Pipeline Manager error message.

2. Determine any new suspense subreasons.
3. Edit the *BRM_home***/sys/msgs/suspense_reason_code.en_US** file by adding suspense reasons as strings and mapping them to integers.

   For example, this shows a sample entry for a suspense reason with an ID of 1:

   ```
   STR
         ID = 1 ;
         VERSION = 1 ;
         STRING - "Unable to identify customer information" ;
   END
   ```

   For example, this shows a sample entry for a suspense subreason with an ID of 2:

   ```
   DOMAIN = "suspense_subreason_1" ;
   STR
         ID = 2;
         VERSION = 1 ;
         STRING = "B number missing" ;
   END
   ```

> **Note:**
>
> - The default string has an ID of 0. This string appears by default in the case of an error not mapped to a suspense reason.
> - The reason code numbers 65535 and 65534 are reserved for use by BRM.

The format of the **suspense_reason_code.***locale* file is similar to that of the **reasons.locale** file.

4. Map your suspense reasons to Pipeline Manager error messages by editing the **pin_suspense_reason_code** file (for CDRs) or the **pin_batch_suspense_reason_code** file (for CDR files) in the *BRM_home***/sys/data/config** directory.

   For example, this shows the default entries in the **pin_suspense_reason_code** file:

```
#ErrorCodes         SuspenseReason       SuspenseSubReason

# Default error (error that is not specified in this file)
00000                    0                   0

# Framework errors
00464                    1                   1
00479                    1                   1
00496                    1                   1
00497                    1                   1
00480                    1                   1
00481                    1                   1
00482                    1                   1

00208                    9                   0
00209                    7                   0
```

5. Load your localized strings into the database by using the **load_localized_strings** utility. See "load_localized_strings" in *BRM Developer's Guide* for the utility's syntax and parameters.

> **Note:**
>
> - The **load_localized_strings** utility requires a configuration file to function correctly.
> - If you're loading a localized version of this file, use the correct file extension for your locale.

```
load_localized_strings filename.locale
```

   where *filename.locale* is the name and location of the file that contains the localized strings.

6. Load your suspense reason code mapping into the database by using the **load_pin_suspense_reason_code** or **load_pin_batch_suspense_reason_code** utility (in the *BRM_home***/bin** directory). For details, see "load_pin_suspense_reason_code" or "load_pin_batch_suspense_reason_code" in *BRM Suspending and Recycling Event Records*.

> **Note:**
>
> The **load_pin_suspense_reason_code** and **load_pin_batch_suspense_reason_code** utilities require configuration files to function correctly.

Example syntax for CDRs:

```
load_pin_suspense_reason_code  pin_suspense_reason_code
```

Example syntax for CDR files:

```
load_pin_batch_suspense_reason_code  pin_batch_suspense_reason_code
```

7. Verify that the strings were loaded by displaying the **/strings** objects using the Object Browser or the **robj** command with the **testnap** utility.
8. Stop and restart the Connection Manager (CM).
9. Stop and restart Suspense Management Center.

Your suspense reason and subreason strings are now loaded into the BRM database to be displayed and used by Suspense Management Center.

Configuring Pipeline Manager for Suspense Manager

The following Pipeline Manager configuration steps are required for Suspense Manager:

1. Configure a standard recycling pipeline. The Suspense Manager Pipeline Manager configuration builds on the steps used to create a standard recycling pipeline. See "Configuring Standard Recycling".
2. Configure a rating pipeline with the following pipeline modules:
   - Configuring FCT_PreSuspense
   - Configuring SuspenseCreateOutput
3. Configure a pre-recycling pipeline. See "Configuring a Pre-Recycling Pipeline".

Configuring FCT_PreSuspense

You added FCT_PreSuspense as the first function module of the pipeline during standard recycling configuration. For details, see "Configuring Standard Recycling".

The FCT_PreSuspense registry requires additional configuration for Suspense Manager, such as for the object class definition you created in "Adding /suspended_usage Subclasses with Queryable Fields".

The example FCT_PreSuspense registry below shows queryable fields for the **/suspended_usage/telco**, **/suspended_usage/telco/gsm**, and **/suspended_usage/telco/gprs** objects.

```
PreSuspense
   {
      ModuleName = FCT_PreSuspense
      Module
      {
         Active = True
         QueryableFields
      {
         # table name. If more than one table, use a separate block
         SUSP_USAGE_TELCO_INFO_T
         {
            # format : <database_column_name> = <edr_conatiner_field_name>
            BYTES_IN       = DETAIL.VOLUME_RECEIVED
            BYTES_OUT      = DETAIL.VOLUME_SENT
            CALLED_TO      = DETAIL.B_NUMBER
            #CALLING_FROM  = DETAIL.B_NUMBER
            CALL_DURATION  = DETAIL.DURATION
            PRIMARY_MSID   = DETAIL.A_NUMBER
            SERVICE_TYPE   = DETAIL.BASIC_SERVICE
            START_TIME     = DETAIL.CHARGING_START_TIMESTAMP
            USAGE_TYPE     = DETAIL.USAGE_TYPE
         }
         SUSP_USAGE_TELCO_GPRS_INFO_T
         {
            # format : <database_column_name> = <edr_conatiner_field_name>
            APN            = DETAIL.ASS_GPRS_EXT.APN_ADDRESS
            GGSN_ADDRESS   = DETAIL.ASS_GPRS_EXT.GGSN_ADDRESS
            NODE_ID        = DETAIL.ASS_GPRS_EXT.NODE_ID
            SECONDARY_MSID = DETAIL.ASS_GPRS_EXT.PORT_NUMBER
            SGSN_ADDRESS   = DETAIL.ASS_GPRS_EXT.SGSN_ADDRESS
         }
            SUSP_USAGE_TELCO_GSM_INFO_T
            {
               APN             = DETAIL.ASS_GSMW_EXT.APN_ADDRESS
               CELL_ID         = DETAIL.ASS_GSMW_EXT.CELL_ID
               DESTINATION_SID = DETAIL.ASS_GSMW_EXT.TERMINATING_SWITCH_IDENTIFICATION
               DIALED_NUMBER   = DETAIL.ASS_GSMW_EXT.DIALED_DIGITS
               ORIGIN_SID      = DETAIL.ASS_GSMW_EXT.ORIGINATING_SWITCH_IDENTIFICATION
               SECONDARY_MSID  = DETAIL.ASS_GSMW_EXT.PORT_NUMBER
            }
         }
      }
   }
}
```

For details about the syntax, see "FCT_PreSuspense" in *BRM Pipeline Manager Reference*.

Configuring SuspenseCreateOutput

Configure the SuspenseCreateOutput module as one of the output modules for this pipeline.

You can use the sample output module in "Configuring Standard Recycling" in a rating pipeline with one change. You need to change the **EventType** entry from **/suspended_usage** to **/suspended_usage/***type*, where *type* is the subclass you created in "Adding /suspended_usage Subclasses with Queryable Fields".

This example shows the **/suspended_usage/telco** being used:

```
SuspenseCreateOutput
   {
      ModuleName = OUT_GenericStream
      EventType  = /suspended_usage/telco
      ...
   }
```

Configuring a Pre-Recycling Pipeline

You configured a pre-recycling pipeline as part of standard recycling configuration.

The pre-recycling pipeline uses the INP_Recycle module, which reads suspended usage records from the BRM database, restores original EDRs, applies edits, and pushes EDRs into the pre-recycling pipeline.

For Suspense Manager, the INP_Recycle module does the following:

- Sets the process status to recycling or test recycling, depending on the processing mode selected in Suspense Management Center.
- Sets override reasons in the DETAIL.ASS_SUSPENSE_EDT.OVERRIDE_REASONS field, and the batch ID to DETAIL.ORIGINAL_BATCH_ID.
- Gives feedback to DAT_Recycle about the status of recycling (commit, cancel, or rollback).
- Takes the original batch ID (from a routing switch, mediation system, or other source) from the /suspended_usage object and copies it to DETAIL.ORIGINAL_BATCH_ID. It also creates a new batch ID for each batch of recycled records and sets it in the HEADER.BATCH_ID and DETAIL.BATCH_ID fields of those records. FCT_PreSuspense appends DETAIL.BATCH_ID with more information to ensure that it remains unique.

To configure a pre-recycling pipeline, see "Configuring a Pre-Recycling Pipeline". Also, perform these additional steps, which are required for Suspense Manager:

1. In the INP_Recycle module, change this EXT_InEasyDB entry:

   ```
   SqlDetail     = StdRecycleDetail.sql
   ```

   to this:

   ```
   SqlDetail     = RecycleDetail.sql
   ```

   EXT_InEasyDB uses this file to read queryable fields in **/suspended_usage** objects.
2. Edit the *Pipeline_home***/database/Oracle/Scripts/Suspense/RecycleDetail.sql** script.
   - Add any custom queryable fields you added in "Adding /suspended_usage Subclasses with Queryable Fields".
   - Remove any non-editable fields from the SELECT statement to improve performance.

Setting Up Suspended Event (SE) Loader for Suspense Manager

This section explains the configuration steps to load suspended CDRs into the BRM database. You used the steps in "Configuring SE Loader for Standard Recycling" to configure the SE Loader for standard recycling.

To configure your **Infranet.properties** file and store your Suspense Manager customizations in **/suspended_usage** objects:

1. Append the contents of **suspense_Infranet.properties** to your **Infranet.properties** file:

   ```
   cd BRM_home/apps/pin_rel
   cat suspense_Infranet.properties Infranet.properties
   ```
2. Edit the *BRM_home***/apps/pin_rel/Infranet.properties** file to create new event types for each of your **/suspended_usage** or **/suspended_batch** subclasses and for temporary objects. Use the **/suspended_usage/telco** section of *BRM_home***/apps/pin_rel/ Infranet.properties** as a guide.

   You use the queryable fields you set up in "Selecting a List of Queryable EDR Fields" in this step.

Setting Up Suspended Batch (SB) Loader for Suspense Manager

To configure SB Loader to load CDR files in the BRM database:

1. Add FCT_BatchSuspense as the first functional module of the pre-processing pipeline. See "FCT_BatchSuspense" in *BRM Pipeline Manager Reference*.

   > **Note:**
   >
   > You can add this module to any pipeline doing file-level validation, but this is most likely the pre-processing pipeline.

   Specify the object you will use to store suspended CDR file information using the **StorableClass** registry entry in FCT_BatchSuspense. The default object is **/suspended_batch/cdr**.

2. Add these entries to your CM's **pin.conf** file to provide a connection to the database to store your suspended CDR files:

   ```
   - nap cm_ptr ip_host port_no
   - nap login_type 1
   - - userid 0.0.0.1 /service/pcm_client 1
   - nap login_name root.0.0.0.1
   - nap login_pw password
   ```

Creating Indexes for Search Templates

By default, Suspense Manager does not include any database indexes for searches other than indexes based on POIDs. You can improve database performance by creating indexes for your most common searches. The example below guides you through the process.

> **Note:**
>
> If there are many indexes on the tables for **/suspended_usage** objects, you run the risk of degrading SE Loader performance during bulk loading of **/suspended_usage** objects. Experiment to find the right balance of indexes for your system.

Example search template:

```
#Suspense Management Template
#Fri Nov 14 09:16:53 PST 2003
PIN_FLD_CALL_DURATION.max=
PIN_FLD_SUSPENSE_REASON.value=<All>
PIN_FLD_CALL_DURATION.selected=false
PIN_FLD_EDITED.value=<All suspended>
PIN_FLD_TEST_SUSPENSE_SUBREASON.value=<All>
PIN_FLD_RECORD_TYPE.selected=true
PIN_FLD_FILENAME.selected=true
PIN_FLD_TEST_ERROR_CODE.min=
PIN_FLD_STATUS.value=Suspended
PIN_FLD_RECORD_TYPE.text=
PIN_FLD_PIPELINE_NAME.text= datadictionary.class=/suspended_usage/telco
```

```
PIN_FLD_PIPELINE_ERROR_CODE.max= PIN_FLD_TEST_SUSPENSE_SUBREASON.selected=false
PIN_FLD_SUSPENSE_SUBREASON.selected=false
PIN_FLD_SERVICE_CODE.text=
PIN_FLD_NUM_RECYCLES.max=0
PIN_FLD_PIPELINE_NAME.selected=false
PIN_FLD_SUSPENSE_REASON.selected=true
PIN_FLD_TEST_SUSPENSE_REASON.value=<All>
PIN_FLD_START_TIME.selected=false
PIN_FLD_CALLING_FROM.text=
PIN_FLD_CALL_DURATION.min=
PIN_FLD_NUM_RECYCLES.selected=true
PIN_FLD_FILENAME.text=
PIN_FLD_EDITED.enabled=true
PIN_FLD_CALLED_TO.selected=true
PIN_FLD_STATUS.selected=false
PIN_FLD_TEST_SUSPENSE_REASON.selected=false
PIN_FLD_RECYCLE_T.selected=false
PIN_FLD_TEST_ERROR_CODE.selected=false
PIN_FLD_CALLING_FROM.selected=true
PIN_FLD_CALLED_TO.text=
PIN_FLD_TEST_ERROR_CODE.max=
PIN_FLD_BATCH_ID.selected=false
PIN_FLD_BATCH_ID.text=
PIN_FLD_PIPELINE_ERROR_CODE.min=
PIN_FLD_SERVICE_CODE.selected=false
PIN_FLD_EDITED.selected=false
PIN_FLD_SUSPENSE_SUBREASON.value=<All>
PIN_FLD_NUM_RECYCLES.min=0
PIN_FLD_PIPELINE_ERROR_CODE.selected=true
```

The example search template translates into this SQL statement:

```
SQL> select st.called_to, st.calling_from, s.filename, s.error_code,
SQL> s.suspense_reason, s.num_recycles from suspended_usage_t s,
SQL> susp_usage_telco_info_t st where s.status = 0 and s.num_recycles
SQL> >= 0 and s.num_recycles <= 0 and s.poid_id0 = st.obj_id0;
```

For Oracle databases, use the statements below to determine which indexes would improve performance. To evaluate this SQL statement, turn on autotrace and run this statement.

This is the output:

```
SQL> set autotrace on;
SQL> select st.called_to, st.calling_from, s.filename, s.error_code,
SQL> s.suspense_reason, s.num_recycles from suspended_usage_t s,
SQL> susp_usage_telco_info_t st where s.status = 0 and s.num_recycles
SQL> >= 0 and s.num_recycles <= 0 and s.poid_id0 = st.obj_id0;
...
13 rows selected.

Execution Plan
----------------------------------------------------------
   0        SELECT STATEMENT Optimizer=CHOOSE
   1    0   NESTED LOOPS
   2    1   TABLE ACCESS (FULL) OF 'SUSP_USAGE_TELCO_INFO_T'
   3    1   TABLE ACCESS (BY INDEX ROWID) OF 'SUSPENDED_USAGE_T'
   4    3   INDEX (UNIQUE SCAN) OF 'I_SUSPENDED_USAGE__ID' (UNIQUE)

Statistics
----------------------------------------------------------
         176  recursive calls
           0  db block gets
          38  consistent gets
          0  physical reads
          0  redo size
       1218  bytes sent via SQL*Net to client
        430  bytes received via SQL*Net from client
          2  SQL*Net roundtrips to/from client
          4  sorts (memory)
          0  sorts (disk)
         13  rows processed
```

The Execution Plan lists TABLE ACCESS (FULL), indicating that search performance would be better if you had created the indexes. Based on the select statement, add appropriate indexes. In this example, add them to **num_recycles** and the status in **suspended_usage_t**. This sample statement creates those indexes:

```
SQL> create index i_susp_usage_test on suspended_usage_t (status,
SQL> num_recycles);
```

After creating the indexes, rerunning the select statement results in a more efficient Execution Plan:

```
Execution Plan
----------------------------------------------------------
   0        SELECT STATEMENT Optimizer=CHOOSE
   1    0   NESTED LOOPS
   2    1   TABLE ACCESS (BY INDEX ROWID) OF 'SUSPENDED_USAGE_T'
   3    2   INDEX (RANGE SCAN) OF 'I_SUSP_USAGE_TEST' (NON-UNIQUE)
   4    1   TABLE ACCESS (BY INDEX ROWID) OF 'SUSP_USAGE_TELCO_INFO_T'
   5    4   INDEX (UNIQUE SCAN) OF 'I_SUSP_USAGE_TELCO__ID' (UNIQUE)

Statistics
----------------------------------------------------------
         0  recursive calls
         0  db block gets
        19  consistent gets
         0  physical reads
         0  redo size
      1218  bytes sent via SQL*Net to client
       430  bytes received via SQL*Net from client
         2  SQL*Net roundtrips to/from client
         0  sorts (memory)
         0  sorts (disk)
        13  rows processed
```

The table scan does not read FULL this time, and there are no **recursive calls** and fewer **consistent gets**. The result is a more efficient call.

Configuring and Customizing Suspense Management Center

To configure and customize the Suspense Management Center client application, add your custom fields to Suspense Management Center and Suspense Management Center Web Start. For more information, see:

- Adding Custom Fields to Suspense Management Center
- Adding Custom Fields to Suspense Management Center Web Start

Adding Custom Fields to Suspense Management Center

Edit **custom.properties** in the *BRM_home***/Program Files/Portal Software/SuspenseManagementCenter/lib** directory, adding any custom fields from your **/**

suspended_usage subclasses. Suspense Management Center displays the fields defined in the **custom.properties** files.

This example defines a new field called **Record Type:** to display in Suspense Management Center:

```
#1. Specify the display name:
#
#field.<dd_field_name>.name = <display name>

field.PIN_FLD_RECORD_TYPE.name = Record Type:
```

Adding Custom Fields to Suspense Management Center Web Start

Edit the **SuspenseManager_en.jnlp** file (in the *Web_Start_home* directory), adding any custom fields from your **/suspended_usage** subclasses. The web version of Suspense Management Center displays the fields that are defined in **SuspenseManager_en.jnlp**.

This example defines a new field called **Record Type:** to display in Suspense Management Center:

```
<resources>
      <j2se version="1.4*"/>
      ...
      ...
      <jar href="lib/Suspense_Management_Help_en.jar"/>
      <extension name="JavaHelp" href="3plibs/jsoft.jnlp"/>
      <property name="suspensemanagement.home" value="f:/apache/apache2/htdocs" />
      <property name="field.PIN_FLD_RECORD_TYPE.name" value="Record Type:" />
      <property name="field.PIN_FLD_RECORD_TYPE.column" value="Record Type" />
</resources>
```

Configuring Event Notification for Suspense Manager

When suspended EDRs are recycled or written off, Suspense Manager uses event notification to call opcodes that perform the appropriate follow-up operations.

Although any subclass of the **/event** class can be used to trigger event notification, Suspense Manager generates the following events specifically to use for event notification:

- **/event/notification/suspense/recycle**: When this event occurs, the EAI framework publishing opcode is called by default.
- **/event/notification/suspense/writeoff**: When this event occurs, PCM_OP_PROCESS_AUDIT_CREATE_WRITEOFF_SUMMARY is called be default.
- **/event/notification/suspense/delete**: When this event occurs, *no* opcode is called by default.
- **/event/notification/suspense/edit**: When this event occurs, *no* opcode is called by default.

Before you can use Suspense Manager, you must configure the event notification feature as follows:

1. If your system has multiple configuration files for event notification, merge them.
2. Ensure that the merged file includes entries for these events:
    - (For Revenue Assurance Manager only) From *BRM_home***/sys/data/config/pin_notify_ra**:

/event/notification/suspense/writeoff

- From *BRM_home*/**sys/data/config/pin_notify_ifw_sync**:

  /event/notification/suspense/recycle

3. (Optional) Add entries for these events to your final event notification list:
   - **/event/notification/suspense/edit**
   - **/event/notification/suspense/delete**

> **Note:**
>
> These events are *not* in a default event notification configuration file. You must manually add them to your final event notification list.

4. (Optional) If necessary, add, modify, or delete entries in your final event notification list to accommodate your business needs.
5. (Optional) If necessary, create custom code for event notification to trigger to accommodate your business needs.
6. Load your final event notification list into the BRM database.

Configuring Debugging (Optional)

You can optionally configure Suspense Management Center to write debugging information to log files and the Suspense Management Center console.

Suspense Management Center provides the following ways for capturing and displaying debugging information:

- The **SuspenseManagementCenter_opcodes.log** log file captures all opcode input and output flists used by Suspense Management Center.
- The **javapcm.log** file contains detailed debugging information. By default, the logging level is set to **0**, the lowest level. The highest level is **3**. You must set the error buffer to **true** to enable **javapcm** logging.
- The **Dloglevel** entry creates a console window for the Suspense Management Center that displays error messages and debugging information.

To configure debugging in Suspense Management Center:

1. Set up Java PCM Logging by adding these entries to the **Infranet.properties** file (in the *BRM_home*/**Program Files/Portal Software/SuspenseManagementCenter/lib** directory):

   ```
   infranet.log.level=3
   infranet.log.logallebuf=true
   infranet.log.opcodes.enabled=true
   infranet.log.opcodes.file=SuspenseManagementCenter_opcodes.log
   ```

2. Set up the Suspense Management Center console to display logging information by opening the **RunSM.bat** file (in the *BRM_home*/**Program Files/Portal Software/SuspenseManagementCenter/lib** directory), changing the **javaw** entry to **java**, and adding this parameter:

   ```
   java -Dloglevel="ALL".
   ```

If you complete these configuration steps, logging information is displayed in the Suspense Management Center console window and these log files:

- *BRM_home*/**Program Files/Portal Software/SuspenseManagementCenter/lib/javapcm.log**
- *BRM_home*/**Program Files/Portal Software/SuspenseManagementCenter/lib/SuspenseManagementCenter_opcodes.log**

Configuring the Number of Suspended Records to Process in a Transaction

To configure the number of suspended records you want the suspense management operations to process in a transaction, perform the following tasks:

1. Edit the *BRM_home*/**sys/data/config/pin_suspense_params** file to specify the maximum number of records to process in a transaction. The file includes examples and instructions.
2. Load the contents of the file into the **/config/suspense_params** object in the BRM database by using **load_pin_suspense_params**. See "load_pin_suspense_params" in *BRM Suspending and Recycling Event Records*.

Suspense Management Center and the **pin_recycle** utility read the **/config/suspense_params** file to get the number of records to process in each opcode call and determine the number of times to call the opcodes. For more information, see "Processing Suspended Records in Multiple Steps" in *BRM Opcode Guide*.

Suspense Management Center Permission Types

Table 6-5 shows the permissions you can set for Suspense Management Center.

Table 6-5 Suspense Management Center Permission Types

Permission Type	Provides Permission To . . .	Max Value Applies
/appcenter/suspensemgt	Use Suspense Management Center.	No
/appcenter/suspensemgt/archive_and_purge	Archive and purge call records, save the records to an archive file, and remove them from the BRM database.	Yes
/appcenter/suspensemgt/batch	Search batches.	Yes
/appcenter/suspensemgt/batch_writeoff	Write off batches.	Yes
/appcenter/suspensemgt/batch_purge	Remove batches from the database.	Yes
/appcenter/suspensemgt/batch_resubmit	Resubmit batches and send them back through a pipeline for rating.	Yes
/appcenter/suspensemgt/bulkedit	Edit a large number of suspended call records in one database operation.	Yes
/appcenter/suspensemgt/bulkpurge	Delete a large number of suspended call records in one database operation.	Yes
/appcenter/suspensemgt/bulkrecycle	Recycle a large number of suspended call records in one database operation.	Yes
/appcenter/suspensemgt/bulkwriteoff	Write off a large number of suspended call records in one database operation.	Yes
/appcenter/suspensemgt/edit	Edit suspended call records.	Yes

Table 6-5   (Cont.) Suspense Management Center Permission Types

Permission Type	Provides Permission To . . .	Max Value Applies
/appcenter/suspensemgt/purge	Purge call records from the BRM database.	Yes
/appcenter/suspensemgt/records	Search records.	Yes
/appcenter/suspensemgt/recycle	Recycle suspended call records and send them back through a pipeline for rating.	Yes
/appcenter/suspensemgt/restore	Restore call records from an archive file.	No
/appcenter/suspensemgt/undo_edit	Undo edits to suspended call records.	No
/appcenter/suspensemgt/writeoff	Write off suspended call records.	Yes

7

Customizing Suspense Manager

You can customize Oracle Communications Billing and Revenue Management (BRM) Suspense Manager to meet your business needs.

Topics in this document:

- Processing a Large Number of Suspended Records
- Overriding Pipeline Suspense Handling Rules
- Using Suspense Management Center with Standard Recycling Call Records
- About Suspense Manager Performance

Processing a Large Number of Suspended Records

You can define search criteria to edit, delete, recycle, and write off a large number of suspended records. In Suspense Management Center, you define the search criteria for a specific action, such as edit, recycle, write off, or delete. Suspense Manager opcodes then perform the specified action on all records that meet the search criteria and are in a valid state for the action. For more information about the opcodes, see "Suspense Management" in *BRM Opcode Guide*.

To avoid a large database transaction during bulk operations, you can specify the number of records to process in each transaction in bulk operations. Based on the number you specify, Suspense Management Center and the **pin_recycle** utility perform several transactions to process the records in the search result. See "Configuring the Number of Suspended Records to Process in a Transaction".

CSRs who perform operations on large numbers of records require specific permissions to perform these operations.

Overriding Pipeline Suspense Handling Rules

During recycling, Suspense Management Center lets you process call records and call record batches that do not pass your pipeline validation rules. This override feature allows you to capture and temporarily hold suspicious calls in a suspended state until you can inspect them.

If they pass inspection, you can override your validation rules and recycle the call records to capture the revenue they represent. You select the override reasons from the Suspense Management Center Recycle screen. The suspense reasons are then overridden for all records in the recycle CDR file. This directs Suspense Manager to successfully process the individual CDRs or CDR files, even though they do not pass your pipeline validation rules.

> **Note:**
>
> The suspense reasons for individual CDRs are separate from those of individual CDRs and must be handled separately.

Changing the List of Override Reasons

The override reasons offered to Customer Service Representatives (CSRs) during recycling is configurable.

To change the list of override reasons:

1. Edit the *BRM_home***/sys/data/config/pin_suspense_override_reason** file.
2. Go to the *BRM_home***/bin** directory.
3. Load the **pin_suspense_override_reason** file into your database:

   ```
   load_pin_suspense_override_reason  pin_suspense_override_reason
   ```

> **Note:**
>
> The **load_pin_suspense_override_reason** utility requires a configuration file to function correctly.

For information about the utility's syntax and parameters, see "load_pin_suspense_override_reason" in *BRM Suspending and Recycling Event Records*.

Changing the List of CDR File Override Reasons

The CDR file override reasons offered to CSRs while resubmitting them are configurable and is separate from the override reasons for individual call records.

To change the list of CDR file override reasons:

1. Edit the *BRM_home***/sys/data/config/pin_batch_suspense_override_reason** file.
2. Go to the *BRM_home***/bin** directory.
3. Load the **pin_batch_suspense_override_reason** file into your database:

   ```
   load_pin_batch_suspense_override_reason  pin_batch_suspense_override_reason
   ```

> **Note:**
>
> The utility requires a configuration file to function correctly.

For information about the utility's syntax and parameters, see "load_pin_batch_suspense_override_reason" in *BRM Suspending and Recycling Event Records*.

Using Suspense Management Center with Standard Recycling Call Records

If you upgraded from standard recycling to Suspense Manager, your database contains two types of call records.

- Call records created using standard recycling use the default **/suspended_usage** fields and have a suspense reason of **Other**.
- Records created under Suspense Manager have a **type** that corresponds to your custom **/suspended_usage** subclasses and have the suspense reasons you created during the Suspense Manager installation and configuration process.

The records you created using standard recycling can be recycled, written off, and deleted using Suspense Management Center. To search for *all* records created under standard recycling, search for call records with a:

- Type of **/suspended_usage**
- Suspense reason of **Other**

To limit the search further, enter the values for any of the **/suspended_usage** fields used by standard recycling.

Standard recycling does not produce CDR file suspense records.

About Suspense Manager Performance

The more call records you attempt to edit, recycle, delete, archive, or archive then delete, the longer it takes. It is impossible to say exactly how long because every implementation is different, but 30,000 records will take a few minutes, and recycling 300,000 records will take many minutes.

8
Recycling EDRs in Pipeline-Only Systems

EDR recycling is used by systems that use Pipeline Manager but do not store suspended EDRs in the Oracle Communications Billing and Revenue Management (BRM) database. It uses FCT_PreRecycle to mark EDRs for recycling, and FCT_Recycle to send the rejected EDRs to a file for manual processing.

Topics in this document:

- About Recycling EDRs
- How the FCT_Reject Module Works
- How the FCT_PreRecycle Module Works
- How the FCT_Recycle Module Works
- Recycling EDRs in Test Mode
- Recycling EDRs

Systems using BRM with Pipeline Manager use either the standard recycling tools or the Suspense Manager service integration component for recycling and deleting EDRs. For details, see "About the EDR Recycling Features".

> **Note:**
>
> Before reading this document, you should be familiar with how Pipeline Manager works and how to configure it.

About Recycling EDRs

When processing a CDR file, there might be non-valid EDRs in the file, or your pipelines might not be set up correctly to handle certain EDRs. You use EDR recycling to fix configuration problems and re-process EDRs.

The recycling process uses these pipeline modules:

- FCT_Reject
- FCT_PreRecycle
- FCT_Recycle

EDR recycling consists of the following high-level steps:

1. You start Pipeline Manager with active FCT_PreRecycle, FCT_Recycle, and FCT_Reject modules. (The FCT_PreRecycle and FCT_Recycle modules do nothing until you begin the recycling process using a semaphore.)
2. When an EDR is processed, a module may find an error in it. The module appends the error to the EDR and sets a flag to indicate that it has an error. The EDR is then sent to the next module. If any more errors are found, each module adds them.

3. The FCT_Reject module analyzes the errors in the EDR. If necessary, it moves the EDR to a reject file.
4. You examine the errors and determine how to reconfigure Pipeline Manager to prevent the errors.
5. You use a semaphore file entry to start the pre-recycling process, which sends the rejected EDRs through the pipeline again. The FCT_PreRecycle module adds a flag to the EDR to let the other modules know that it is being recycled.

   You can pre-recycle and recycle EDRs in test mode or real mode. Typically, you run the pre-recycle and recycling processes in test mode to see if the errors have been fixed. When there are no longer any errors, you pre-recycle and recycle in real mode.
6. The FCT_Recycle module runs at the end of the pipeline. It does one of the following:
   - In test mode, the module creates a report about the processing but does not send the EDRs to an output file.
   - In recycle mode, the module sends the results to an output file and attaches a sequence number to it.

> **Note:**
>
> You can configure the output module to send an entire file to the error directory if it contains many errors. You can also configure the threshold for the number of errors allowed per file. See "Specifying the Maximum Errors Allowed in an Input File" in *BRM Setting Up Pipeline Rating and Discounting*.

How the FCT_PreRecycle Module Works

The FCT_PreRecycle module is always the first module in the pipeline. Although you can *activate* the FCT_PreRecycle module from the startup registry, you cannot run the FCT_PreRecycle module from the startup registry. You must *run* it by using a semaphore file.

The FCT_PreRecycle module does the following:

1. Gets the file of rejected EDRs from the reject output stream directory.
2. Puts the rejected EDR file into the input directory for recycling. It uses the same input directory as the incoming CDR files. It adds a recycle suffix to the file and a sequence number, so the original input file in the output directory cannot be overwritten.

   You can recycle all EDRs in the reject directory or list specific files to recycle. See "Recycling EDRs".
3. For each EDR to recycle, sets a value in the INTERN_PROCESS_STATUS field to indicate that it is being recycled. This tells the FCT_Recycle module which EDRs to process, and allows the discounting modules to recalculate discount amounts correctly.
   - The value is set to **1** if the EDR is being recycled.
   - The value is set to **2** if the recycling is in test mode.

You can recycle all EDRs in the reject directory or list specific files to recycle.

How the FCT_Reject Module Works

Run the FCT_Reject module after all rating and enrichment modules and as the second-to-last function module in the pipeline. (The last function module is the FCT_Suspense module). You run it second-to-last because all potential errors must be found before the FCT_Reject module processes the EDRs.

You can run the FCT_Reject module from the registry or by using a semaphore file entry.

The FCT_Reject module does the following:

1. The FCT_Reject module checks the error status of the EDR. If the EDR contains an error status with a **warning** or **critical** severity, the EDR is rejected. The FCT_Reject module changes the value of the DETAIL.DISCARDING field from 0 to 1.

   > **Note:**
   >
   > If the DETAIL.DISCARDING field is already set to 1, the EDR was rejected in a previous pass through the pipeline, and is rejected again.

   If the error type in the EDR is not identified in the registry **StreamMap** entry, the EDR is sent to default reject stream.

2. By default, the EDR is moved to the reject stream, as identified in the **UseRejectStream** registry entry. The EDR is stored in a file that is used by the recycling modules. EDRs can also be rejected in the recycle process.

   If the reject stream is not specified, the EDR is moved to the normal output stream, but the discard field is set to **1**, indicating that the EDR has been rejected.

How the FCT_Recycle Module Works

The FCT_Recycle module is the last function module in the pipeline, before the output.

You activate the FCT_Recycle module from the startup registry, but it does nothing until the FCT_PreRecycle module starts the recycling process.

The FCT_Recycle module reads the INTERN_PROCESS_STATUS field for each EDR.

- If the value is **2**, recycling is in test mode. The FCT_Recycle module doesn't send the EDRs to an output directory. Instead, the FCT_Recycle module creates a report with the following data:
  - Stream name.
  - Total number of processed EDRs.
  - Number of EDRs that can be recycled without an error.
  - Number of EDRs that still generate an error.
  - List of all errors.
  - The wholesale charge amount from all successfully recycled EDRs. (This data is taken from the WHOLESALE_CHARGE field.)
  - The wholesale charge amount from all EDRs that still have errors. (This data is taken from the WHOLESALE_CHARGE field.)

  - The total duration for all successfully recycled EDRs. (This data is taken from the DURATION_MINUTES field.)
  - The total duration from all EDRs that still have errors. (This data is taken from the DURATION_MINUTES field.)

  You can use this data to determine if the EDRs are worth further configuration and processing.

- If the value is **1**, recycling occurs. All EDRs are processed as usual, with the following differences in comparison to normal input file processing:
  - A sequence number is generated.
  - The sequence offset value is generated.
  - The sequence check is inactivated.

Recycling EDRs in Test Mode

After determining that EDRs have been rejected, you correct any pipeline problems that caused the problem. You then recycle the CDR file in test mode to ensure that the changes have had the desired affect.

To test recycle EDRs:

1. Configure the FCT_Reject module. See "Configuring FCT_Reject to Reject EDRs".

   Typically, rejected EDRs are sent to the reject stream. You configure the reject stream in the registry in the following places:

   - In the FCT_Reject module pipeline configuration
   - In the Output stream

     For a sample output stream configuration, see "Sample Output Configuration" in *BRM Pipeline Manager Reference*.

2. Configure the FCT_PreRecycle module. See "Configuring FCT_PreRecycle to Move Rejected EDRs to the Input Directory".

   You configure FCT_PreRecycle in the following places:

   - In the FCT_PreRecycle module pipeline configuration
   - In the input stream

   The module uses the same input configuration as the incoming CDR files, so you don't need to configure a separate input stream.

3. Configure the FCT_Recycle module. See "Configuring FCT_Recycle to Recycle or Test Recycle EDRs".

   Configure the FCT_Recycle module's **RecycleLog** registry entry to specify the message file parameters. These settings are specified in the **ProcessLog** registry section. For more information, see "LOG" in *BRM Pipeline Manager Reference*.

4. Use a semaphore to inactivate the FCT_Reject module:

   ```
   Module
   Reject.Module.Active = False
   ```

5. Use a semaphore to run the FCT_PreRecycle module in test mode:

   ```
   Recycle.Module.RecycleTest {}
   ```

6. Review the log files that you configured in FCT_Recycle for errors, and repeat these steps as necessary.

Recycling EDRs

When you finish test recycling EDRs, follow these steps to do the actual recycling:

1. Configure the FCT_Reject module. See "Configuring FCT_Reject to Reject EDRs".

   Typically, rejected EDRs are sent to the reject output stream. You configure the reject output stream in the following places:

   - In the FCT_Reject module pipeline configuration
   - In the Output stream

     For a sample output stream configuration, see "Sample Output Configuration" in *BRM Pipeline Manager Reference*.

2. Configure the FCT_PreRecycle module. See "Configuring FCT_PreRecycle to Move Rejected EDRs to the Input Directory".

   You configure the reject stream in the registry in the following places:

   - In the FCT_PreRecycle module pipeline configuration
   - In the input stream

   The module uses the same input configuration as the incoming CDR files, so you don't need to configure a separate input stream.

3. Configure the FCT_Recycle module. See "Configuring FCT_Recycle to Recycle or Test Recycle EDRs".

   Configure the FCT_Recycle module's **RecycleLog** registry entry to specify the message file parameters. You specify these settings in the **ProcessLog** registry section. For more information, see "LOG" in *BRM Pipeline Manager Reference*.

4. Use a semaphore to run the FCT_PreRecycle module:

   - To recycle all files:

     ```
     ifw.Pipelines.PRE_RECYCLE.Functions.Processing.FunctionPool.PreRecycle.Module.Rec
     ycle {}
     ```

   - To recycle only specific files:

     ```
     ifw.Pipelines.PRE_RECYCLE.Functions.Processing.FunctionPool.PreRecycle.Module.Rec
     ycle.
     File = ./format_a/abc.cdr
     ```

5. Review the log files that you configured in FCT_Reject for errors.

Configuring FCT_PreRecycle to Move Rejected EDRs to the Input Directory

Table 8-1 shows how to configure the FCT_PreRecycle pipeline module to retrieve a file of rejected EDRs, add a recycle suffix to the file name, and move the file to the recycling pipeline's input directory.

Table 8-1 Configuring FCT_PreRecycle

Registry Entry	Description	Required Value
Active	Specifies if the module is active or inactive.	**True**
RecycleSuffix	Specifies the suffix for the file that contains the EDRs that need recycling. The suffix is automatically appended when the file is moved from the reject directory to the input directory. If it is empty, no suffix is added.	Suffix
RecyFileName	Specifies the file name and path for the file that contains the EDRs that need recycling.	FileName

For example, this FCT_PreRecycle registry configuration specifies to:

- Save the EDRs that need recycling to the **recycle.dat** file
- Add the **RecycleFile** suffix when moving files containing EDRs that need recycling to the input directory

```
PreRecycle
{
   ModuleName = FCT_PreRecycle
   Module
   {
      Active        = True
      RecycleSuffix = RecycleFile
      RecyFileName  = ./recycle.dat
   }
}
```

Configuring FCT_Reject to Reject EDRs

Table 8-2 shows how to configure the FCT_Reject pipeline module to determine whether to reject EDRs. If the EDR is rejected, the module sends it to the specified reject output stream.

Table 8-2 Configuring FCT_Reject for Recycling EDRs

Registry Entry	Description	Required Value
Active	Specifies if the module is active or inactive.	**True**
NotifyonReject	Specifies whether to notify other modules that an EDR has been rejected.	**True** or **False**
UseRejectStream	Specifies whether to send rejected EDRs to the reject output stream.	**True**
CallAssemblingModule	If you use both the FCT_CallAssembling module and the FCT_Reject module in a pipeline, use the CallAssemblingModule registry entry to ensure that the complete EDRs are recycled. Otherwise, only part of the EDR is recycled.	Do not set or specify the registry path to FCT_CallAssembling

Table 8-2 (Cont.) Configuring FCT_Reject for Recycling EDRs

Registry Entry	Description	Required Value
MinErrorSeverity	Specifies to reject EDRs that have a specified severity, allowing EDRs to be processed with warning or normal error messages without being rejected. For recycling EDRs in pipeline-only systems, set this entry to one of these values: • **3** = Minor • **4** = Major • **5** = Critical To allow warning and normal messages without rejecting the EDR, set this entry to **3**.	**3**, **4**, or **5**
StreamMap	Specifies a list of error types mapped to output streams. To create • One reject output stream, list only one BRM error message to output stream mapping. • Multiple reject output streams, list multiple BRM error messages each mapped to a separate reject output stream. For a list of BRM error messages, see "EDR Suspense Reasons".	BRMErrorMessage = OutputStreamName

For example, this FCT_Reject registry configuration specifies to:

- Notify other modules that an EDR has been rejected.
- Use the FCT_CallAssembling module.
- Reject EDRs with minor, major, and critical error messages.
- Send rejected EDRs to one of three output streams: RapOutput, CiberOutput, and TadigOutput.

```
Reject
{
   ModuleName = FCT_Reject
   {
      Active                 = True
      NotifyOnReject         = True
      UseRejectStream        = True
      CallAssemblingModule =
ifw.Pipelines.Pipe.Functions.Standard.FunctionPool.CallAssembling
      MinErrorSeverity       = 3
      StreamMap
      {
         ERR_TAP3_RET         = RapOutput
         ERR_CIBER_RET        = CiberOutput
         ERR_TADIG_NOT_FOUND = TadigOutput
      }
   }
}
```

Configuring FCT_Recycle to Recycle or Test Recycle EDRs

Table 8-3 shows how to configure the FCT_Reject pipeline module to:

- (In test mode) Create a report with data about the processing but does not send the EDR to an output file.
- (In recycle mode) Send the EDR file to an output stream and attach a sequence number the file.

Table 8-3   Configuring FCT_Recycle for Recycling EDRs

Registry Entry	Description
Active	Specifies if the module is active or inactive.
RecycleLog.MessageFilePath	Specifies the path where the log file can find the message database.
RecycleLog.MessageFilePrefix	Specifies the prefix for collecting the files from the message file path.
RecycleLog.MessageFileSuffix	Specifies the suffix for collecting the files from the message file path.
RecycleLog.FilePath	Specifies the path in which the log file is written.
RecycleLog.FilePrefix	Specifies the prefix for the log file.
RecycleLog.FileSuffix	Specifies the suffix for the log file.

For example, this FCT_Recycle registry configuration specifies to:

```
Recycle
{
   ModuleName = FCT_Recycle
   Module
   {
      Active = True
      RecycleLog
      {
         MessageFilePath   = ..
         MessageFilePrefix = Framework
         MessageFileSuffix = msg
         FilePath          = ../tmp/log01
         FilePrefix        = rej_
         FileSuffix        = .log
      }
   }
}
```

9 EDR Suspense Reasons

Learn about the default reasons for suspending event data records (EDRs) in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

- About Suspense Reasons
- Pipeline Manager Suspense Reasons
- DAT_Account Suspense Reasons
- FCT_AggreGate Suspense Reasons
- FCT_BalancePlugIn Suspense Reasons
- FCT_CallAssembling Suspense Reasons
- FCT_CustomerRating Suspense Reasons
- FCT_Discount Suspense Reasons
- FCT_EnhancedSplitting Suspense Reasons
- FCT_ExchangeRate Suspense Reasons
- FCT_MainRating Suspense Reasons
- FCT_PreRating Suspense Reasons
- FCT_Tadig2PlmnPlugIn Suspense Reasons
- CIBER Suspense Reasons
- Tap Suspense Reasons
- FCT_AccountRouter Suspense Reasons
- FCT_TriggerBill Suspense Reasons
- FCT_Account Suspense Reasons
- FCT_Filter_Set Suspense Reasons

About Suspense Reasons

This chapter shows the default mapping between various BRM error messages and Suspense Manager reasons for suspending event data records (EDRs). The information in this table is derived from several different source files and is much easier to understand in this format.

You can change this mapping or add your own error messages as appropriate for your BRM implementation. For information on how to add to or change these messages, see "Changing the List of Suspense Reasons and Subreasons".

Pipeline Manager Suspense Reasons

Table 9-1 shows the default mapping between BRM error messages and Suspense Manager reasons for suspending event data records (EDRs).

Table 9-1 Pipeline Manager Errors

BRM Error Message	Suspense Reason String	Suspense Subreason String
ERR_INPUT_MAPPING_FAILED	Batch rating engine processing error	Record mapping error
ERR_TRANSFER_CUTOFF_VIOLATED	Batch rating engine processing error	Record mapping error
ERR_INCORRECT_FILLER_LENGTH	Batch rating engine processing error	Record mapping error
ERR_CANNOT_JOIN_EVENT_HANDLER_PROC	Batch rating engine processing error	Record mapping error
ERR_INVALID_RECORD_NUMBER	Batch rating engine processing error	Record mapping error
ERR_INVALID_FIRST_CALL_TIMESTAMP	Batch rating engine processing error	Record mapping error
ERR_INVALID_LAST_CALL_TIMESTAMP	Batch rating engine processing error	Record mapping error
ERR_A_CUSTOMER_NOT_FOUND	Customer data error	Customer error
ERR_NO_SPLITTING_PERFORMED	Splitting error	Splitting error
ERR_ADD_DATABLOCK	Batch rating engine processing error	Batch rating engine processing error
ERR_DATABASE	Batch rating engine processing error	Database error
ERR_RATEPLAN_NOT_FOUND	Rating error	Invalid charge
ERR_INSUFFICIENT_MEMORY	Batch rating engine processing error	Insufficient memory

DAT_Account Suspense Reasons

Table 9-2 shows the default mapping between BRM error messages and Suspense Manager reasons when the DAT_Account module suspends EDRs.

Table 9-2 DAT_Account Errors

BRM Error Message	Suspense Reason String	Suspense Subreason String
ERR_BALANCE_GROUP_UPDATE	Customer data error	Internal error
ERR_BILL_INFO_UPDATE	Customer data error	Internal error
ERR_MAPPING_TABLE_UPDATE	Customer data error	Internal error
ERR_CUSTOMER_LOGIN_NOT_FOUND	Customer data error	Customer error
ERR_CUSTOMER_LOGIN_SERVICE_NOT_FOUND	Customer data error	Customer error
ERR_CUSTOMER_LOGIN_ACCOUNT_NOT_FOUND	Customer data error	Customer error
ERR_SERVICE_NOT_CONFIGURED	Customer data error	Customer error
ERR_CUSTOMER_SERVICE_NOT_FOUND	Customer data error	Customer error
ERR_CUSTOMER_EDR_PARSING	Customer data error	Customer error
ERR_CUSTOMER_LOGIN_NOT_VALID_FOR_TIME	Customer data error	Customer error
ERR_CUSTOMER_NO_VALID_PRODUCT	Customer data error	Customer error

Table 9-2 (Cont.) DAT_Account Errors

BRM Error Message	Suspense Reason String	Suspense Subreason String
ERR_CUSTOMER_NO_VALID_PRODUCT_RATING	Customer data error	Customer error
ERR_CUSTOMER_INVALID_ITEM_POID	Customer data error	Customer error
ERR_INVALID_OUTPUT_STREAM	Customer data error	Customer error
ERR_CUSTOMER_LOGIN_INTERNAL_ERROR	Customer data error	Internal error
ERR_ACCRT_MESSAGE	Customer data error	Internal error
ERR_NOT_IMPLEMENTED	Customer data error	Internal error

FCT_AggreGate Suspense Reasons

Table 9-3 shows the default mapping between various BRM error messages and Suspense Manager reasons for suspending EDRs.

Table 9-3 FCT_AggreGate Errors

BRM Error Messages	Suspense Reason String	Suspense Subreason String
ERR_NO_DEPENDENT_CLASS_DEFINED	Record aggregation error	Aggregation error
ERR_EDR_ITERATOR_FAILURE	Record aggregation error	Aggregation error

FCT_BalancePlugIn Suspense Reasons

Table 9-4 shows the default mapping between various BRM error messages and Suspense Manager reasons for suspending EDRs.

Table 9-4 FCT_BalancePlugIn Errors

BRM Error Messages	Suspense Reason String	Suspense Subreason String
ERR_BALANCE_NOT_FOUND	Billing record error	Failed to add discount balance info

FCT_CallAssembling Suspense Reasons

Table 9-5 shows the default mapping between various BRM error messages and Suspense Manager reasons for suspending EDRs.

Table 9-5 FCT_CallAssembling Errors

BRM Error Messages	Suspense Reason String	Suspense Subreason String
ERR_CHAIN_REFERENCE_MISSING	Call assembling error	Call assembling error
ERR_INVALID_STATE_INDICATOR	Call assembling error	Call assembling error

Table 9-5 (Cont.) FCT_CallAssembling Errors

BRM Error Messages	Suspense Reason String	Suspense Subreason String
ERR_REJECTED_EDR_NOT_IN_WORKFILE	Call assembling error	Call assembling error
ERR_EDR_ALREADY_CLOSED	Call assembling error	Call assembling error

FCT_CustomerRating Suspense Reasons

Table 9-6 shows the default mapping between various BRM error messages and Suspense Manager reasons for suspending EDRs.

Table 9-6 FCT_CustomerRating Errors

BRM Error Messages	Suspense Reason String	Suspense Subreason String
ERR_CUSTOMER_NOT_FOUND	Rating error	Customer rating error
ERR_RATEPLAN_NOT_DEFINED	Rating error	Customer rating error

FCT_Discount Suspense Reasons

Table 9-7 shows the default mapping between BRM error messages and Suspense Manager reasons for EDRs suspended by the FCT_Discount module.

Table 9-7 FCT_Discount Errors

BRM Error Messages	Suspense Reason String	Suspense Subreason String
ERR_INVALID_GRANT_TYPE	Discount processing error	Discount processing error
ERR_PLUGIN_INVALID_STATE	Discount processing error	Discount processing error
ERR_EDR_ITERATOR	Discount processing error	Discount processing error
ERR_EDRPACK_NOT_READY_DSC	Discount processing error	Discount processing error
ERR_BEGIN_EDR	Discount processing error	Discount processing error
ERR_COMMIT_EDR	Discount processing error	Discount processing error
ERR_CANCEL_EDR	Discount processing error	Discount processing error
ERR_ROLLBACK_EDR	Discount processing error	Discount processing error
ERR_CANCEL_DEMANDED_EDR	Discount processing error	Discount processing error
ERR_BEGIN_DSC_TRANSACTION	Discount processing error	Discount processing error
ERR_END_DSC_TRANSACTION	Discount processing error	Discount processing error
ERR_DSC_CONF_NOT_FOUND	Discount processing error	Discount processing error
ERR_INVALID_THRESHOLD_TYPE	Discount processing error	Discount processing error
ERR_INVALID_DISCOUNT_TYPE	Discount processing error	Discount processing error
ERR_DISCOUNT_DETACH_MODULE	Discount processing error	Discount processing error
ERR_ACCOUNT_COMMIT_RESTART	Discount processing error	Discount processing error
ERR_ACCOUNT_PREPARECOMMIT	Discount processing error	Discount processing error

Table 9-7 (Cont.) FCT_Discount Errors

BRM Error Messages	Suspense Reason String	Suspense Subreason String
ERR_ACCOUNT_COMMIT	Discount processing error	Discount processing error
ERR_ACCOUNT_CANCEL	Discount processing error	Discount processing error
ERR_ACCOUNT_ROLLBACK	Discount processing error	Discount processing error
ERR_NO_ACCOUNT_BALANCE_ERROR	Discount processing error	Discount processing error
ERR_CANNOT_COMPILE_SCRIPT	Discount processing error	Discount processing error
ERR_CURRENCY_RESID_NOT_FOUND	Discount processing error	Discount processing error
ERR_INVALID_BASE_AMOUNT	Discount processing error	Discount processing error
ERR_INVALID_BASE_EXPR	Discount processing error	Discount processing error
ERR_INVALID_COND_AMOUNT	Discount processing error	Discount processing error
WRN_INVALID_DRUM_AMOUNT	Discount processing error	Discount processing error
ERR_INVALID_THRESHOLD_AMOUNT	Discount processing error	Discount processing error
ERR_REJECT_EDR	Discount processing error	Discount processing error
ERR_EXPR_REF_CP	Discount processing error	Discount processing error
ERR_GETTING_BG_ID	Discount processing error	Discount processing error

FCT_EnhancedSplitting Suspense Reasons

Table 9-8 shows the default mapping between various BRM error messages and Suspense Manager reasons for suspending EDRs.

Table 9-8 FCT_EnhancedSplitting Errors

BRM Error Messages	Suspense Reason String	Suspense Subreason String
ERR_NO_SPLITTING_ENTRY	Splitting error	Splitting error

FCT_ExchangeRate Suspense Reasons

Table 9-9 shows the default mapping between various BRM error messages and Suspense Manager reasons for suspending EDRs.

Table 9-9 FCT_ExchangeRate Errors

BRM Error Messages	Suspense Reason String	Suspense Subreason String
ERR_EXCHANGERATE_BRK_HEADERDATE	Rating error	Exchange rate error
ERR_EXCHANGERATE_FILEDATE_NOT_EXIST	Rating error	Exchange rate error
ERR_EXCHANGERATE_NOT_FOUND	Rating error	Exchange rate error

FCT_MainRating Suspense Reasons

Table 9-10 shows the default mapping between various BRM error messages and Suspense Manager reasons for suspending EDRs.

Table 9-10 FCT_MainRating Errors

BRM Error Messages	Suspense Reason String	Suspense Subreason String
ERR_RATEPLAN_VERSION_ID_NOT_FOUND	Rating error	Invalid charge
ERR_RATEPLAN_VERSION_DATE_NOT_FOUND	Rating error	Invalid charge
ERR_TIMEMODEL_NOT_FOUND	Rating error	Invalid time model or time zone
ERR_RATE_PRICEMODEL_NOT_FOUND	Rating error	Invalid pricing
ERR_TIMEZONE_NOT_FOUND	Rating error	Invalid time model or time zone
ERR_PRICEMODEL_NOT_FOUND	Rating error	Invalid pricing
ERR_PRICEMODEL_RUM_NOT_FOUND	Rating error	Invalid pricing
ERR_PRICEMODEL_STEP_NOT_FOUND	Rating error	Invalid pricing
ERR_PRICEMODEL_CONFIG_NOT_FOUND	Rating error	Invalid pricing
ERR_INVALID_ADDON_TYPE	Rating error	Other main rating error
ERR_RUM_GROUP_NOT_FOUND	Rating error	Other main rating error

FCT_PreRating Suspense Reasons

Table 9-11 shows the default mapping between various BRM error messages and Suspense Manager reasons for suspending EDRs.

Table 9-11 FCT_PreRating Errors

BRM Error Messages	Suspense Reason String	Suspense Subreason String
ERR_RATEPLAN_NOT_A_NUMBER	Rating error	Invalid charge
ERR_RATEPLAN_TYPE_INV	Rating error	Invalid charge
ERR_RATEPLAN_VERSION_NOT_FOUND	Rating error	Invalid charge
ERR_NO_RATEPLAN	Rating error	Invalid charge

FCT_Tadig2PlmnPlugIn Suspense Reasons

Table 9-12 shows the default mapping between various BRM error messages and Suspense Manager reasons for suspending EDRs.

Table 9-12 FCT_Tadig2PlmnPlugIn Errors

BRM Error Messages	Suspense Reason String	Suspense Subreason String
ERR_TADIG_NOT_FOUND	Mapping problem	TADIG to PLMN map error

CIBER Suspense Reasons

Table 9-13 shows the default mapping between various BRM error messages and Suspense Manager reasons for suspending EDRs.

Table 9-13 CIBER Errors

BRM Error Messages	Suspense Reason String	Suspense Subreason String
ERR_CIBER_RET	Roaming records error	CIBER record error

Tap Suspense Reasons

Table 9-14 shows the default mapping between various BRM error messages and Suspense Manager reasons for suspending EDRs.

Table 9-14 Tap Errors

BRM Error Messages	Suspense Reason String	Suspense Subreason String
ERR_TAP3_RET	Roaming records error	TAP record error

FCT_AccountRouter Suspense Reasons

Table 9-15 shows the default mapping between various BRM error messages and Suspense Manager reasons for suspending EDRs.

Table 9-15 FCT_AccountRouter Errors

BRM Error Messages	Suspense Reason String	Suspense Subreason String
ERR_JOB_AMT_MIGRATION	Internal suspension	Account being migrated

FCT_TriggerBill Suspense Reasons

Table 9-16 shows the default mapping between various BRM error messages and Suspense Manager reasons for suspending EDRs.

Table 9-16 FCT_TriggerBill Errors

BRM Error Messages	Suspense Reason String	Suspense Subreason String
ERR_TRIGGER_BILLING	Internal suspension	Awaiting billing of account

FCT_Account Suspense Reasons

Table 9-17 shows the default mapping between various BRM error messages and Suspense Manager reasons for suspending EDRs.

Table 9-17 FCT_Account Errors

BRM Error Messages	Suspense Reason String	Suspense Subreason String
ERR_JOB_RERATING_ACCOUNT	Internal Suspension	Account usage being re-rated

FCT_Filter_Set Suspense Reasons

Table 9-18 shows the default mapping between various BRM error messages and Suspense Manager reasons for suspending event data records (EDRs).

Table 9-18 FCT_Filter_Set Errors

BRM Error Messages	Suspense Reason String	Suspense Subreason String
ERR_INVALID_DISCOUNT_ID	Filter set error	Invalid Discount ID
ERR_INVALID_PRODUCT_ID	Filter set error	Invalid Product ID

10 Troubleshooting Suspense and Recycle Issues

Learn how to fix common problems that you may encounter when suspending and recycling pipeline event data records (EDRs) in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

- Increasing Heap Size to Avoid Performance Problems
- Unexpected Log Messages Caused by Missing MaxErrorRates Entry

Increasing Heap Size to Avoid Performance Problems

If the searches you run in Suspense Management Center return particularly large results, your performance may slow noticeably, or you may get "Out of memory" error messages. The solution is to increase your maximum heap size. The exact amount varies significantly with your needs and system resources. If performance is poor or you get "Out of memory" messages frequently, start by doubling the maximum heap size to 128 MB. Remember, however, that making the heap size too large will degrade the performance of other processes.

Depending on whether you have a standalone or a WebStart BRM implementation, there are two ways to increase the maximum heap size.

To increase the heap size for standalone implementations:

1. Edit the *BRM_home***/lib/runSMC.bat** file to increase the heap size (memory allocation pool) to solve "Out of memory" messages.

   By default, Suspense Management Center has a maximum heap size of 64 MB. This variable is controlled by the **-Xmx** *size* entry in the Suspense Manager Center startup line in **runSMC.bat**. No **-Xmx** *size* entry is present in the startup line by default. To increase the heap size, add this entry and a number (in MBs) to the Suspense Management Center startup line.

   This example adds a 128 MB maximum heap size to Suspense Management Center:

```
@start C:\PROGRA~1\COMMON~1\PORTAL~1\JRE\bin\javaw.exe -Xmx128m -cp ".;%SMCDIR%;%SMCDIR%\lib;%SMCDIR%
\lib\suspensemgtmain.jar;%SMCDIR%\lib\pfc.jar;%SMCDIR%\3plibs\jh.jar;%SMCDIR%\lib\pcmext.jar;%SMCDIR%
\lib\pcm.jar;%SMCDIR%\lib\Suspense_Management_Help_en.jar;%SMCDIR%
\lib\Application_Center_Help_en.jar;" com.portal.appcenter.AppCenterMain suspensemgtsuite
```

   > **Note:**
   >
   > Be sure to precede and follow the **-Xmx** *size* entry with a space.

2. Stop and restart Suspense Management Center to make the change take effect.

To increase the heap size for WebStart implementations:

1. Open your **SuspenseManagement_***locale***.jnlp** file.

2. Change the **j2se** element to include a max-heap-size attribute.

   The default entry looks like this:

   ```
   <j2se version="1.4*"/>
   ```

   This sample entry changes the maximum heap size to 128 MBs:

   ```
   <j2se version="1.4*" max-heap-size="128m"/>
   ```

   > **Note:**
   >
   > The maximum heap size specified in the JNLP file is used for all associated Suspense Management Center clients.

3. Stop and restart Suspense Management Center to make the change take effect.

Unexpected Log Messages Caused by Missing MaxErrorRates Entry

Your pipeline output section must contain a **MaxErrorRates** registry section with at least one entry. If this entry is missing, your log files include a misleading message like this one:

```
"16.11.2030 21:00:37 All checks are successful. File can be recycled."
```